J Glob Optim
DOI 10.1007/s10898-013-0035-4

Algorithms for nonnegative matrix and tensor factorizations: a unified view based on block coordinate descent framework

Jingu Kim · Yunlong He · Haesun Park

Received: 5 September 2012 / Accepted: 17 January 2013

Abstract We review algorithms developed for nonnegative matrix factorization (NMF) and nonnegative tensor factorization (NTF) from a unified view based on the block coordinate descent (BCD) framework. NMF and NTF are low-rank approximation methods for matrices and tensors in which the low-rank factors are constrained to have only nonnegative elements. The nonnegativity constraints have been shown to enable natural interpretations and allow better solutions in numerous applications including text analysis, computer vision, and bioinformatics. However, the computation of NMF and NTF remains challenging and expensive due the constraints. Numerous algorithmic approaches have been proposed to efficiently compute NMF and NTF. The BCD framework in constrained non-linear optimization readily explains the theoretical convergence properties of several efficient NMF and NTF algorithms, which are consistent with experimental observations reported in literature. In addition, we discuss algorithms that do not fit in the BCD framework contrasting them from those based on the BCD framework. With insights acquired from the unified perspective, we also propose efficient algorithms for updating NMF when there is a small change in the reduced dimension or in the data. The effectiveness of the proposed updating algorithms are validated experimentally with synthetic and real-world data sets.

Keywords Nonnegative matrix factorization · Nonnegative tensor factorization · Low-rank approximation · Block coordinate descent

J. Kim
Nokia Inc., 200 S. Mathilda Ave, Sunnyvale, CA, USA
e-mail: jingu.kim@nokia.com

Y. He
School of Mathematics, Georgia Institute of Technology, Atlanta, GA, USA
e-mail: heyunlong@gatech.edu

H. Park (✉)
School of Computational Science and Engineering, Georgia Institute of Technology, Atlanta, GA, USA
e-mail: hpark@cc.gatech.edu

1 Introduction

Nonnegative matrix factorization (NMF) is a dimension reduction and factor analysis method. Many dimension reduction techniques are closely related to the low-rank approximations of matrices, and NMF is special in that the low-rank factor matrices are constrained to have only nonnegative elements. The nonnegativity reflects the inherent representation of data in many application areas, and the resulting low-rank factors lead to physically natural interpretations [66]. NMF was first introduced by Paatero and Tapper [74] as positive matrix factorization and subsequently popularized by Lee and Seung [66]. Over the last decade, NMF has received enormous attention and has been successfully applied to a broad range of important problems in areas including text mining [77,85], computer vision [47,69], bioinformatics [10,23,52], spectral data analysis [76], and blind source separation [22] among many others.

Suppose a nonnegative matrix $\mathbf{A} \in \mathbb{R}^{M\times N}$ is given. When the desired lower dimension is K, the goal of NMF is to find two matrices $\mathbf{W} \in \mathbb{R}^{M\times K}$ and $\mathbf{H} \in \mathbb{R}^{N\times K}$ having only nonnegative elements such that

$$\mathbf{A} \approx \mathbf{W}\mathbf{H}^T. \tag{1}$$

According to (1), each data point, which is represented as a column in $\mathbf{A}$, can be approximated by an additive combination of the nonnegative basis vectors, which are represented as columns in $\mathbf{W}$. As the goal of dimension reduction is to discover compact representation in the form of (1), K is assumed to satisfy that $K < \min\{M, N\}$. Matrices $\mathbf{W}$ and $\mathbf{H}$ are found by solving an optimization problem defined with Frobenius norm, Kullback-Leibler divergence [67,68], or other divergences [24,68]. In this paper, we focus on the NMF based on Frobenius norm, which is the most commonly used formulation:

$$\begin{aligned} &\min_{\mathbf{W},\mathbf{H}} f(\mathbf{W},\mathbf{H}) = \|\mathbf{A} - \mathbf{W}\mathbf{H}^T\|_F^2 \qquad (2)\\ &\text{subject to } \mathbf{W} \geq 0, \mathbf{H} \geq 0. \end{aligned}$$

The constraints in (2) mean that all the elements in $\mathbf{W}$ and $\mathbf{H}$ are nonnegative. Problem (2) is a non-convex optimization problem with respect to variables $\mathbf{W}$ and $\mathbf{H}$, and finding its global minimum is NP-hard [81]. A good algorithm therefore is expected to compute a local minimum of (2).

Our first goal in this paper is to provide an overview of algorithms developed to solve (2) from a unifying perspective. Our review is organized based on the block coordinate descent (BCD) method in non-linear optimization, within which we show that most successful NMF algorithms and their convergence behavior can be explained. Among numerous algorithms studied for NMF, the most popular is the multiplicative updating rule by Lee and Seung [67]. This algorithm has an advantage of being simple and easy to implement, and it has contributed greatly to the popularity of NMF. However, slow convergence of the multiplicative updating rule has been pointed out [40,71], and more efficient algorithms equipped with stronger theoretical convergence property have been introduced. The efficient algorithms are based on either the alternating nonnegative least squares (ANLS) framework [53,59,71] or the hierarchical alternating least squares (HALS) method [19,20]. We show that these methods can be derived using one common framework of the BCD method and then characterize some of the most promising NMF algorithms in Sect. 2. Algorithms for accelerating the BCD-based methods as well as algorithms that do not fit in the BCD framework are summarized in Sect. 3, where we explain how they differ from the BCD-based methods. In the ANLS method, the subproblems appear as the nonnegativity constrained least squares (NLS) problems. Much research has been devoted to design NMF algorithms based on efficient methods to solve the NLS subproblems [18,42,51,53,59,71]. A review of many successful algorithms for the NLS subproblems is provided in Sect. 4 with discussion on their advantages and disadvantages.

Extending our discussion to low-rank approximations of tensors, we show that algorithms for some nonnegative tensor factorization (NTF) can similarly be elucidated based on the BCD framework. Tensors are mathematical objects for representing multidimensional arrays; vectors and matrices are first-order and second-order special cases of tensors, respectively. The canonical decomposition (CANDECOMP) [14] or the parallel factorization (PARAFAC) [43], which we denote by the CP decomposition, is one of the natural extensions of the singular value decomposition to higher order tensors. The CP decomposition with nonnegativity constraints imposed on the loading matrices [19,21,32,54,60,84], which we denote by nonnegative CP (NCP), can be computed in a way that is similar to the NMF computation. We introduce details of the NCP decomposition and summarize its computation methods based on the BCD method in Sect. 5.

Lastly, in addition to providing a unified perspective, our review leads to the realizations of NMF in more dynamic environments. Such a common case arises when we have to compute NMF for several K values, which is often needed to determine a proper K value from data. Based on insights from the unified perspective, we propose an efficient algorithm for updating NMF when K varies. We show how this method can compute NMFs for a set of different K values with much less computational burden. Another case occurs when NMF needs to be updated efficiently for a data set which keeps changing due to the inclusion of new data or the removal of obsolete data. This often occurs when the matrices represent data from time-varying signals in computer vision [11] or text mining [13]. We propose an updating algorithm which takes advantage of the fact that most of data in two consecutive time steps are overlapped so that we do not have to compute NMF from scratch. Algorithms for these cases are discussed in Sect. 7, and their experimental validations are provided in Sect. 8.

Our discussion is focused on the algorithmic developments of NMF formulated as (2). In Sect. 9, we only briefly discuss other aspects of NMF and conclude the paper.

Notations: Notations used in this paper are as follows. A lowercase or an uppercase letter, such as x or X, denotes a scalar; a boldface lowercase letter, such as $\mathbf{x}$, denotes a vector; a boldface uppercase letter, such as $\mathbf{X}$, denotes a matrix; and a boldface Euler script letter, such as $\mathcal{X}$, denotes a tensor of order three or higher. Indices typically start from 1 to its uppercase letter: For example, $n \in \{1, \dots, N\}$. Elements of a sequence of vectors, matrices, or tensors are denoted by superscripts within parentheses, such as $\mathbf{X}^{(1)}, \dots, \mathbf{X}^{(N)}$, and the entire sequence is denoted by $\left\{\mathbf{X}^{(n)}\right\}$. When matrix $\mathbf{X}$ is given, $(\mathbf{X})_{\cdot i}$ or $\mathbf{x}_{\cdot i}$ denotes its ith column, $(\mathbf{X})_{i\cdot}$ or $\mathbf{x}_{i\cdot}$ denotes its ith row, and x_{ij} denotes its (i, j)th element. For simplicity, we also let $\mathbf{x}_i$ (without a dot) denote the ith column of $\mathbf{X}$. The set of nonnegative real numbers are denoted by $\mathbb{R}_+$, and $\mathbf{X} \geq 0$ indicates that the elements of $\mathbf{X}$ are nonnegative. The notation $[\mathbf{X}]_+$ denotes a matrix that is the same as $\mathbf{X}$ except that all its negative elements are set to zero. A *nonnegative matrix* or a *nonnegative tensor* refers to a matrix or a tensor with only nonnegative elements. The null space of matrix $\mathbf{X}$ is denoted by $null(\mathbf{X})$. Operator $\bigotimes$ denotes element-wise multiplcation of vectors or matrices.

2 A unified view—BCD framework for NMF

The BCD method is a divide-and-conquer strategy that can be generally applied to non-linear optimization problems. It divides variables into several disjoint subgroups and iteratively minimize the objective function with respect to the variables of each subgroup at a time.

We first introduce the BCD framework and its convergence properties and then explain several NMF algorithms under the framework.

Consider a constrained non-linear optimization problem:

$$\min f(\mathbf{x}) \text{ subject to } \mathbf{x} \in \mathcal{X}, \tag{3}$$

where $\mathcal{X}$ is a closed convex subset of $\mathbb{R}^N$. An important assumption to be exploited in the BCD framework is that $\mathcal{X}$ is represented by a Cartesian product:

$$\mathcal{X} = \mathcal{X}_1 \times \cdots \times \mathcal{X}_M, \tag{4}$$

where $\mathcal{X}_m, m = 1, \ldots, M$, is a closed convex subset of $\mathbb{R}^{N_m}$ satisfying $N = \sum_{m=1}^{M} N_m$. Accordingly, vector $\mathbf{x}$ is partitioned as $\mathbf{x} = (\mathbf{x}_1, \ldots, \mathbf{x}_M)$ so that $\mathbf{x}_m \in \mathcal{X}_m$ for $m = 1, \ldots, M$. The BCD method solves for $\mathbf{x}_m$ fixing all other subvectors of $\mathbf{x}$ in a cyclic manner. That is, if $\mathbf{x}^{(i)} = (\mathbf{x}_1^{(i)}, \ldots, \mathbf{x}_M^{(i)})$ is given as the current iterate at the ith step, the algorithm generates the next iterate $\mathbf{x}^{(i+1)} = (\mathbf{x}_1^{(i+1)}, \ldots, \mathbf{x}_M^{(i+1)})$ block by block, according to the solution of the following subproblem:

$$\mathbf{x}_m^{(i+1)} \leftarrow \underset{\xi \in \mathcal{X}_m}{\arg\min}\, f\left(\mathbf{x}_1^{(i+1)}, \ldots, \mathbf{x}_{m-1}^{(i+1)}, \xi, \mathbf{x}_{m+1}^{(i)}, \ldots, \mathbf{x}_M^{(i)}\right). \tag{5}$$

Also known as a *non-linear Gauss-Siedel* method [5], this algorithm updates one block each time, always using the most recently updated values of other blocks $\mathbf{x}_{\tilde{m}}, \tilde{m} \neq m$. This is important since it ensures that after each update the objective function value does not increase. For a sequence $\{\mathbf{x}^{(i)}\}$ where each $\mathbf{x}^{(i)}$ is generated by the BCD method, the following property holds.

Theorem 1 *Suppose f is continuously differentiable in $\mathcal{X} = \mathcal{X}_1 \times \cdots \times \mathcal{X}_M$, where $\mathcal{X}_m, m = 1, \ldots, M$, are closed convex sets. Furthermore, suppose that for all m and i, the minimum of*

$$\min_{\xi \in \mathcal{X}_m} f\left(\mathbf{x}_1^{(i+1)}, \ldots, \mathbf{x}_{m-1}^{(i+1)}, \xi, \mathbf{x}_{m+1}^{(i)}, \ldots, \mathbf{x}_M^{(i)}\right)$$

is uniquely attained. Let $\{\mathbf{x}^{(i)}\}$ be the sequence generated by the BCD method in (5). Then, every limit point of $\{\mathbf{x}^{(i)}\}$ is a stationary point. The uniqueness of the minimum is not required when M is two.

The proof of this theorem for an arbitrary number of blocks is shown in Bertsekas [5], and the last statement regarding the two-block case is due to Grippo and Sciandrone [41]. For a non-convex optimization problem, most algorithms only guarantee the stationarity of a limit point [46,71].

When applying the BCD method to a constrained non-linear programming problem, it is critical to wisely choose a partition of $\mathcal{X}$, whose Cartesian product constitutes $\mathcal{X}$. An important criterion is whether subproblems (5) for $m = 1, \ldots, M$ are efficiently solvable: For example, if the solutions of subproblems appear in a closed form, each update can be computed fast. In addition, it is worth checking whether the solutions of subproblems depend on each other. The BCD method requires that the most recent values need to be used for each subproblem (5). When the solutions of subproblems depend on each other, they have to be computed sequentially to make use of the most recent values; if solutions for some blocks are independent from each other, however, simultaneous computation of them would be possible. We discuss how different choices of partitions lead to different NMF algorithms. Three cases of partitions are shown in Fig. 1, and each case is discussed below.

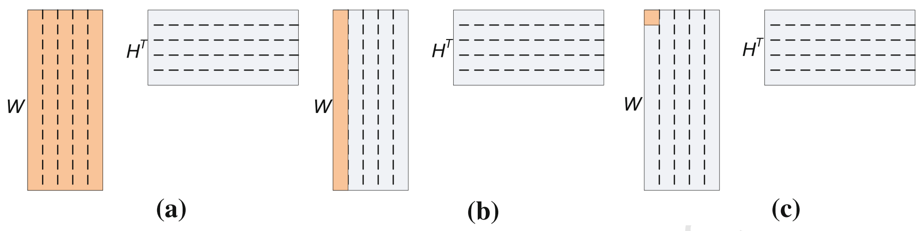

Fig. 1 Different choices of block partitions for the BCD method for NMF where $\mathbf{W} \in \mathbb{R}_+^{M\times K}$ and $\mathbf{H} \in \mathbb{R}_+^{N\times K}$. In each case, the highlighted part for example is updated fixing all the rest. **a** Two matrix blocks. **b** $2K$ vector blocks. **c** $K(M+N)$ scalar blocks

2.1 BCD with two matrix blocks—ANLS method

In (2), a natural partitioning of the variables is the two blocks representing **W** and **H**, as shown in Fig. 1a. In this case, following the BCD method in (5), we take turns solving

$$\mathbf{W} \leftarrow \arg\min_{\mathbf{W}\geq 0} f(\mathbf{W},\mathbf{H}) \text{ and } \mathbf{H} \leftarrow \arg\min_{\mathbf{H}\geq 0} f(\mathbf{W},\mathbf{H}). \tag{6}$$

These subproblems can be written as

$$\min_{\mathbf{W}\geq 0} \|\mathbf{H}\mathbf{W}^T - \mathbf{A}^T\|_F^2 \quad \text{and} \tag{7a}$$

$$\min_{\mathbf{H}\geq 0} \|\mathbf{W}\mathbf{H}^T - \mathbf{A}\|_F^2 \quad . \tag{7b}$$

Since subproblems (7) are the nonnegativity constrained least squares (NLS) problems, the two-block BCD method has been called the alternating nonnegative least square (ANLS) framework [53,59,71]. Even though the subproblems are convex, they do not have a closed-form solution, and a numerical algorithm for the subproblem has to be provided. Several approaches for solving the NLS subproblems proposed in NMF literature are discussed in Sect. 4 [18,42,51,53,59,71]. According to Theorem 1, the convergence property of the ANLS framework can be stated as follows.

Corollary 1 *If a minimum of each subproblem in* (7) *is attained at each step, every limit point of the sequence* $\left\{(\mathbf{W},\mathbf{H})^{(i)}\right\}$ *generated by the ANLS framework is a stationary point of* (2).

Note that the minimum is not required to be unique for the convergence result to hold because the number of blocks are two [41]. Therefore, **H** in (7a) or **W** in (7b) need not be of full column rank for the property in Corollary 1 to hold. On the other hand, some numerical methods for the NLS subproblems require the full rank conditions so that they return a solution that attains a minimum: See Sect. 4 as well as regularization methods in Sect. 2.4.

Subproblems (7) can be decomposed into independent NLS problems with a single right-hand side vector. For example,

$$\min_{\mathbf{W}\geq 0} \left\|\mathbf{H}\mathbf{W}^T - \mathbf{A}^T\right\|_F^2 = \sum_{m=1}^{M} \min_{\mathbf{w}_{m\cdot}\geq 0} \left\|\mathbf{H}\mathbf{w}_{m\cdot}^T - \mathbf{a}_{m\cdot}^T\right\|_F^2, \tag{8}$$

and we can solve the problems in the second term independently. This view corresponds to a BCD method with $M+N$ vector blocks, in which each block corresponds to a row of

W or **H**. In literature, however, this view has not been emphasized because often it is more efficient to solve the NLS problems with multiple right-hand sides altogether: See Sect. 4.

2.2 BCD with $2K$ vector blocks—HALS/RRI method

Let us now partition the unknowns into $2K$ blocks in which each block is a column of **W** or **H**, as shown in Fig. 1b. In this case, it is easier to consider the objective function in the following form:

$$f(\mathbf{w}_1, \ldots, \mathbf{w}_K, \mathbf{h}_1, \ldots, \mathbf{h}_K) = \left\| \mathbf{A} - \sum_{k=1}^{K} \mathbf{w}_k \mathbf{h}_k^T \right\|_F^2, \tag{9}$$

where $\mathbf{W} = [\mathbf{w}_1, \ldots \mathbf{w}_K] \in \mathbb{R}_+^{M\times K}$ and $\mathbf{H} = [\mathbf{h}_1, \ldots, \mathbf{h}_K] \in \mathbb{R}_+^{N\times K}$. The form in (9) represents that **A** is approximated by the sum of K rank-one matrices.

Following the BCD scheme, we can minimize f by iteratively solving

$$\mathbf{w}_k \leftarrow \arg\min_{\mathbf{w}_k \ge 0} f(\mathbf{w}_1, \ldots, \mathbf{w}_K, \mathbf{h}_1, \ldots, \mathbf{h}_K)$$

for $k = 1, \ldots, K$, and

$$\mathbf{h}_k \leftarrow \arg\min_{\mathbf{h}_k \ge 0} f(\mathbf{w}_1, \ldots, \mathbf{w}_K, \mathbf{h}_1, \ldots, \mathbf{h}_K)$$

for $k = 1, \ldots, K$. These subproblems appear as

$$\min_{\mathbf{w}\ge 0} \|\mathbf{h}_k \mathbf{w}^T - \mathbf{R}_k^T\|_F^2 \text{ and } \min_{\mathbf{h}\ge 0} \|\mathbf{w}_k \mathbf{h}^T - \mathbf{R}_k\|_F^2, \tag{10}$$

where

$$\mathbf{R}_k = \mathbf{A} - \sum_{\tilde{k}=1, \tilde{k}\neq k}^{K} \mathbf{w}_{\tilde{k}} \mathbf{h}_{\tilde{k}}^T. \tag{11}$$

A promising aspect of this $2K$ block partitioning is that each subproblem in (10) has a closed-form solution, as characterized in the following theorem.

Theorem 2 *Consider a minimization problem*

$$\min_{\mathbf{v}\ge 0} \|\mathbf{u}\mathbf{v}^T - \mathbf{G}\|_F^2 \tag{12}$$

where $\mathbf{G} \in \mathbb{R}^{M\times N}$ *and* $\mathbf{u} \in \mathbb{R}^M$ *are given. If* **u** *is a nonzero vector,* $\mathbf{v} = \frac{[\mathbf{G}^T\mathbf{u}]_+}{\mathbf{u}^T\mathbf{u}}$ *is the unique solution for* (12)*, where* $([\mathbf{G}^T\mathbf{u}]_+)_n = \max((\mathbf{G}^T\mathbf{u})_n, 0)$ *for* $n = 1, \ldots, N$.

Proof Letting $\mathbf{v}^T = (v_1, \ldots, v_N)$, we have

$$\min_{\mathbf{v}\ge 0} \left\| \mathbf{u}\mathbf{v}^T - \mathbf{G} \right\|_F^2 = \sum_{n=1}^{N} \min_{v_n \ge 0} \|\mathbf{u}v_n - \mathbf{g}_n\|_2^2,$$

where $\mathbf{G} = [\mathbf{g}_1, \ldots, \mathbf{g}_N]$, and the problems in the second term are independent of each other. Let $h(v_n) = \|\mathbf{u}v_n - \mathbf{g}_n\|_2^2 = \|\mathbf{u}\|_2^2 v_n^2 - 2v_n \mathbf{u}^T\mathbf{g}_n + \|\mathbf{g}_n\|_2^2$. Since $\frac{\partial h}{\partial v_n} = 2(v_n\|\mathbf{u}\|_2^2 - \mathbf{g}_n^T\mathbf{u})$, if $\mathbf{g}_n^T\mathbf{u} \ge 0$, it is clear that the minimum value of $h(v_n)$ is attained at $v_n = \frac{\mathbf{g}_n^T\mathbf{u}}{\mathbf{u}^T\mathbf{u}}$. If $\mathbf{g}_n^T\mathbf{u} < 0$, the value of $h(v_n)$ increases as v_n becomes larger than zero, and therefore the minimum is attained at $v_n = 0$. Combining the two cases, the solution can be expressed as $v_n = \frac{[\mathbf{g}_n^T\mathbf{u}]_+}{\mathbf{u}^T\mathbf{u}}$. □

Using Theorem 2, the solutions of (10) can be stated as

$$\mathbf{w}_k \leftarrow \frac{[\mathbf{R}_k \mathbf{h}_k]_+}{\|\mathbf{h}_k\|_2^2} \quad \text{and} \quad \mathbf{h}_k \leftarrow \frac{[\mathbf{R}_k^T \mathbf{w}_k]_+}{\|\mathbf{w}_k\|_2^2}. \tag{13}$$

This $2K$-block BCD algorithm has been studied under the name of the hierarchical alternating least squares (HALS) method by Cichocki et al. [19,20] and the rank-one residue iteration (RRI) independently by Ho [44]. According to Theorem 1, the convergence property of the HALS/RRI algorithm can be written as follows.

Corollary 2 *If the columns of* $\mathbf{W}$ *and* $\mathbf{H}$ *remain nonzero throughout all the iterations and the minimums in* (13) *are attained at each step, every limit point of the sequence* $\left\{(\mathbf{W},\mathbf{H})^{(i)}\right\}$ *generated by the HALS/RRI algorithm is a stationary point of* (2).

In practice, a zero column could occur in $\mathbf{W}$ or $\mathbf{H}$ during the HALS/RRI algorithm. This happens if $\mathbf{h}_k \in null(\mathbf{R}_k)$, $\mathbf{w}_k \in null(\mathbf{R}_k^T)$, $\mathbf{R}_k\mathbf{h}_k \le 0$, or $\mathbf{R}_k^T\mathbf{w}_k \le 0$. To prevent zero columns, a small positive number could be used for the maximum operator in (13): That is, $\max(\cdot,\epsilon)$ with a small positive number ϵ such as 10^{-16} is used instead of $\max(\cdot,0)$ [20,35]. The HALS/RRI algorithm with this modification often shows faster convergence compared to other BCD methods or previously developed methods [37,59]. See Sect. 3.1 for acceleration techniques for the HALS/RRI method and Sect. 6.2 for more discussion on experimental comparisons.

For an efficient implementation, it is not necessary to explicitly compute $\mathbf{R}_k$. Replacing $\mathbf{R}_k$ in (13) with the expression in (11), the solutions can be rewritten as

$$\mathbf{w}_k \leftarrow \left[\mathbf{w}_k + \frac{(\mathbf{AH})_{\cdot k} - (\mathbf{WH}^T\mathbf{H})_{\cdot k}}{(\mathbf{H}^T\mathbf{H})_{kk}}\right]_+ \text{ and} \tag{14a}$$

$$\mathbf{h}_k \leftarrow \left[\mathbf{h}_k + \frac{(\mathbf{A}^T\mathbf{W})_{\cdot k} - (\mathbf{HW}^T\mathbf{W})_{\cdot k}}{(\mathbf{W}^T\mathbf{W})_{kk}}\right]_+. \tag{14b}$$

The choice of update formulae is related with the choice of an update order. Two versions of an update order can be considered:

$$\mathbf{w}_1 \to \mathbf{h}_1 \to \cdots \to \mathbf{w}_K \to \mathbf{h}_K \tag{15}$$

and

$$\mathbf{w}_1 \to \cdots \to \mathbf{w}_K \to \mathbf{h}_1 \to \cdots \to \mathbf{h}_K. \tag{16}$$

When using (13), update order (15) is more efficient because $\mathbf{R}_k$ is explicitly computed and then used to update both $\mathbf{w}_k$ and $\mathbf{h}_k$. When using (14), although either (15) or (16) can be used, update order (16) tends to be more efficient in environments such as MATLAB based on our experience. To update all the elements in $\mathbf{W}$ and $\mathbf{H}$, update formulae (13) with ordering (15) require $8KMN + 3K(M+N)$ floating point operations, whereas update formulae (14) with either choice of ordering require $4KMN + (4K^2 + 6K)(M+N)$ floating point operations. When $K \ll \min(M,N)$, the latter is more efficient. Moreover, the memory requirement of (14) is smaller because $\mathbf{R}_k$ need not be stored. For more details, see Cichocki and Phan [19].

2.3 BCD with $K(M+N)$ scalar blocks

In one extreme, the unknowns can be partitioned into $K(M+N)$ blocks of scalars, as shown in Fig. 1c. In this case, every element of $\mathbf{W}$ and $\mathbf{H}$ is considered as a block in the context of Theorem 1. To this end, it helps to write the objective function as a quadratic function of scalar w_{mk} or h_{nk} assuming all other elements in $\mathbf{W}$ and $\mathbf{H}$ are fixed:

$$f(w_{mk}) = \left\| \left(\mathbf{a}_{m\cdot} - \sum_{\tilde{k}\neq k} w_{m\tilde{k}} \mathbf{h}_{\cdot\tilde{k}}^T \right) - w_{mk}\mathbf{h}_{\cdot k}^T \right\|_2^2 + \text{const}, \tag{17a}$$

$$f(h_{nk}) = \left\| \left(\mathbf{a}_{\cdot n} - \sum_{\tilde{k}\neq k} \mathbf{w}_{\cdot\tilde{k}} h_{n\tilde{k}} \right) - \mathbf{w}_{\cdot k} h_{nk} \right\|_2^2 + \text{const}, \tag{17b}$$

where $\mathbf{a}_{m\cdot}$ and $\mathbf{a}_{\cdot n}$ denote the mth row and the nth column of $\mathbf{A}$, respectively. According to the BCD framework, we iteratively update each block by

$$w_{mk} \leftarrow \operatorname*{arg\,min}_{w_{mk}\geq 0} f(w_{mk}) = \left[w_{mk} + \frac{(\mathbf{AH})_{mk} - (\mathbf{WH}^T\mathbf{H})_{mk}}{(\mathbf{H}^T\mathbf{H})_{kk}} \right]_+ \tag{18a}$$

$$h_{nk} \leftarrow \operatorname*{arg\,min}_{h_{nk}\geq 0} f(h_{nk}) = \left[h_{nk} + \frac{(\mathbf{A}^T\mathbf{W})_{nk} - (\mathbf{HW}^T\mathbf{W})_{nk}}{(\mathbf{W}^T\mathbf{W})_{kk}} \right]_+ . \tag{18b}$$

The updates of w_{mk} and h_{nk} are independent of all other elements in the same column. Therefore, it is possible to update all the elements in each column of $\mathbf{W}$ (and $\mathbf{H}$) simultaneously. Once we organize the update of (18) column-wise, the result is the same as (14). That is, a particular arrangement of the BCD method with scalar blocks is equivalent to the BCD method with $2K$ vector blocks. Accordingly, the HALS/RRI method can be derived by the BCD method either with vector blocks or with scalar blocks. On the other hand, it is not possible to simultaneously solve for the elements in each row of $\mathbf{W}$ (or $\mathbf{H}$) because their solutions depend on each other. The convergence property of the scalar block case is similar to that of the vector block case.

Corollary 3 *If the columns of* $\mathbf{W}$ *and* $\mathbf{H}$ *remain nonzero throughout all the iterations and if the minimums in* (18) *are attained at each step, every limit point of the sequence* $\left\{(\mathbf{W},\mathbf{H})^{(i)}\right\}$ *generated by the BCD method with* $K(M+N)$ *scalar blocks is a stationary point of* (2).

The multiplicative updating rule also uses element-wise updating [67]. However, the multiplicative updating rule is different from the scalar block BCD method in a sense that its solutions are not optimal for subproblems (18). See Sect. 3.2 for more discussion.

2.4 BCD for some variants of NMF

To incorporate extra constraints or prior information into the NMF formulation in (2), various regularization terms can be added. We can consider an objective function

$$\min_{\mathbf{W},\mathbf{H}\geq 0} \left\| \mathbf{A} - \mathbf{WH}^T \right\|_F^2 + \phi(\mathbf{W}) + \psi(\mathbf{H}), \tag{19}$$

where $\phi(\cdot)$ and $\psi(\cdot)$ are regularization terms that often involve matrix or vector norms. Here we discuss the Frobenius-norm and the l_1-norm regularization and show how NMF regularized by those norms can be easily computed using the BCD method. Scalar parameters α or β in this subsection are used to control the strength of regularization.

The Frobenius-norm regularization [53,76] corresponds to

$$\phi(\mathbf{W}) = \alpha \|\mathbf{W}\|_F^2 \ \text{ and } \ \psi(\mathbf{H}) = \beta \|\mathbf{H}\|_F^2 . \tag{20}$$

The Frobenius-norm regularization may be used to prevent the elements of $\mathbf{W}$ or $\mathbf{H}$ from growing too large in their absolute values. It can also be adopted to stabilize the BCD methods. In the two matrix block case, since the uniqueness of the minimum of each subproblem is not required according to Corollary 1, $\mathbf{H}$ in (7a) or $\mathbf{W}$ in (7b) need not be of full column rank. The full column rank condition is however required for some algorithms for the NLS subproblems, as discussed in Sect. 4. As shown below, the Frobenius-norm regularization ensures that the NLS subproblems of the two matrix block case are always defined with a matrix of full column rank. Similarly in the $2K$ vector block or the $K(M+N)$ scalar block cases, the condition that $\mathbf{w}_k$ and $\mathbf{h}_k$ remain nonzero throughout all the iterations can be relaxed when the Frobenius-norm regularization is used.

Applying the BCD framework with two matrix blocks to (19) with the regularization term in (20), $\mathbf{W}$ can be updated as

$$\mathbf{W} \leftarrow \arg\min_{\mathbf{W}\geq 0} \left\| \begin{pmatrix} \mathbf{H} \\ \sqrt{\alpha}\mathbf{I}_K \end{pmatrix} \mathbf{W}^T - \begin{pmatrix} \mathbf{A}^T \\ \mathbf{0}_{K\times M} \end{pmatrix} \right\|_F^2, \tag{21}$$

where $\mathbf{I}_K$ is a $K \times K$ identity matrix and $\mathbf{0}_{K\times M}$ is a $K \times M$ matrix containing only zeros, and $\mathbf{H}$ can be updated with a similar reformulation. Clearly, if α is nonzero, $\begin{pmatrix} \mathbf{H} \\ \sqrt{\alpha}\mathbf{I}_K \end{pmatrix}$ in (21) is of full column rank. Applying the BCD framework with $2K$ vector blocks, a column of $\mathbf{W}$ is updated as

$$\mathbf{w}_k \leftarrow \left[\frac{(\mathbf{H}^T\mathbf{H})_{kk}}{(\mathbf{H}^T\mathbf{H})_{kk}+\alpha}\mathbf{w}_k + \frac{(\mathbf{AH})_{\cdot k} - (\mathbf{WH}^T\mathbf{H})_{\cdot k}}{(\mathbf{H}^T\mathbf{H})_{kk}+\alpha} \right]_+ . \tag{22}$$

If α is nonzero, the solution of (22) is uniquely defined without requiring $\mathbf{h}_k$ to be a nonzero vector.

The l_1-norm regularization can be adopted to promote sparsity in the factor matrices. In many areas such as linear regression [80] and signal processing [16], it has been widely known that the l_1-norm regularization promotes sparse solutions. In NMF, sparsity was shown to improve the part-based interpretation [47] and the clustering ability [52,57]. When sparsity is desired on matrix $\mathbf{H}$, the l_1-norm regularization can be set as

$$\phi(\mathbf{W}) = \alpha\|\mathbf{W}\|_F^2 \quad \text{and} \quad \psi(\mathbf{H}) = \beta\sum_{n=1}^{N}\|\mathbf{h}_{n\cdot}\|_1^2, \tag{23}$$

where $\mathbf{h}_{n\cdot}$ represents the nth row of $\mathbf{H}$. The l_1-norm term of $\psi(\mathbf{H})$ in (23) promotes sparsity on $\mathbf{H}$ while the Frobenius norm term of $\phi(\mathbf{W})$ is needed to prevent $\mathbf{W}$ from growing too large. Similarly, sparsity can be imposed on $\mathbf{W}$ or on both $\mathbf{W}$ and $\mathbf{H}$.

Applying the BCD framework with two matrix blocks to (19) with the regularization term in (23), $\mathbf{W}$ can be updated as (21), and $\mathbf{H}$ can be updated as

$$\mathbf{H} \leftarrow \arg\min_{\mathbf{H}\geq 0} \left\| \begin{pmatrix} \mathbf{W} \\ \sqrt{\beta}\mathbf{1}_{1\times K} \end{pmatrix} \mathbf{H}^T - \begin{pmatrix} \mathbf{A} \\ \mathbf{0}_{1\times N} \end{pmatrix} \right\|_F^2, \tag{24}$$

where $\mathbf{1}_{1\times K}$ is a row vector of length K containing only ones. Applying the BCD framework with $2K$ vector blocks, a column of $\mathbf{W}$ is updated as (22), and a column of $\mathbf{H}$ is updated as

$$\mathbf{h}_k \leftarrow \left[\mathbf{h}_k + \frac{(\mathbf{A}^T\mathbf{W})_{\cdot k} - \mathbf{H}((\mathbf{W}^T\mathbf{W})_{\cdot k} + \beta\mathbf{1}_K)}{(\mathbf{W}^T\mathbf{W})_{kk}+\beta} \right]_+ . \tag{25}$$

Note that the l_1-norm term in (23) is written as the sum of the *squares* of the l_1-norm of the columns of $\mathbf{H}$. Alternatively, we can impose the l_1-norm based regularization without squaring: That is,

$$\phi(\mathbf{W}) = \alpha \|\mathbf{W}\|_F^2 \quad \text{and} \qquad \psi(\mathbf{H}) = \beta \sum_{n=1}^{N} \sum_{k=1}^{K} |\mathbf{h}_{nk}| \,. \tag{26}$$

Although both (23) and (26) promote sparsity, the squared form in (23) is easier to handle with the two matrix block case, as shown above. Applying the $2K$-vector BCD framework on (19) with the regularization term in (26), the update for a column of $\mathbf{h}$ is written as

$$\mathbf{h}_k \leftarrow \left[\mathbf{h}_k + \frac{(\mathbf{A}^T\mathbf{W})_{\cdot k} - (\mathbf{H}\mathbf{W}^T\mathbf{W})_{\cdot k} + \frac{1}{2}\beta \mathbf{1}_K}{(\mathbf{W}^T\mathbf{W})_{kk}} \right]_+ .$$

For more information, see [19], Section 4.7.4 of [22], and Section 4.5 of [44]. When the BCD framework with two matrix blocks is used with the regularization term in (26), a custom algorithm for l_1-regularized least squares problem has to be involved: See, e.g., [30].

3 Acceleration and other approaches

3.1 Accelerated methods

The BCD methods described so far have been very successful for the NMF computation. In addition, several techniques to accelerate the methods have been proposed. Korattikara et al. [62] proposed a subsampling strategy to improve the two matrix block (i.e., ANLS) case. Their main idea is to start with a small factorization problem, which is obtained by random subsampling, and gradually increase the size of subsamples. Under the assumption of asymptotic normality, the decision whether to increase the size is made based on statistical hypothesis testing. Gillis and Glineur [38] proposed a multi-level approach, which also gradually increases the problem size based on a multi-grid representation. The method in [38] is applicable not only to the ANLS methods, but also to the HALS/RRI method and the multiplicative updating method.

Hsieh and Dhillon proposed a greedy coordinate descent method [48]. Unlike the HALS/RRI method, in which every element is updated exactly once per iteration, they selectively choose the elements whose update will lead to the largest decrease of the objective function. Although their method does not follow the BCD framework, they showed that every limit point generated by their method is a stationary point. Gillis and Glineur also proposed an acceleration scheme for the HALS/RRI and the multiplicative updating methods: Unlike the standard versions, their approach repeats updating the elements of $\mathbf{W}$ several times before updating the elements of $\mathbf{H}$ [37]. Noticeable improvements in the speed of convergence is reported.

3.2 Multiplicative updating rules

The multiplicative updating rule [67] is by far the most popular algorithm for NMF. Each element is updated through *multiplications* as

$$w_{mk} \leftarrow w_{mk} \frac{(\mathbf{A}\mathbf{H})_{mk}}{(\mathbf{W}\mathbf{H}^T\mathbf{H})_{mk}}, \;\; h_{nk} \leftarrow h_{nk} \frac{(\mathbf{A}^T\mathbf{W})_{nk}}{(\mathbf{H}\mathbf{W}^T\mathbf{W})_{nk}}. \tag{27}$$

Since elements are updated in this multiplication form, the nonnegativity is always satisfied when $\mathbf{A}$ is nonnegative. This algorithm can be contrasted with the HALS/RRI algorithm as follows. The element-wise gradient descent updates for (2) can be written as

$$w_{mk} \leftarrow w_{mk} + \lambda_{mk}\left[(\mathbf{AH})_{mk} - (\mathbf{WH}^T\mathbf{H})_{mk}\right] \text{ and}$$
$$h_{nk} \leftarrow h_{nk} + \mu_{nk}\left[(\mathbf{A}^T\mathbf{W})_{nk} - (\mathbf{HW}^T\mathbf{W})_{nk}\right],$$

where λ_{mk} and μ_{nk} represent step-lengths. The multiplicative updating rule is obtained by taking

$$\lambda_{mk} = \frac{w_{mk}}{(\mathbf{WH}^T\mathbf{H})_{mk}} \text{ and } \mu_{nk} = \frac{h_{nk}}{(\mathbf{HW}^T\mathbf{W})_{nk}}, \tag{28}$$

whereas the HALS/RRI algorithm interpreted as the BCD method with scalar blocks as in (18) is obtained by taking

$$\lambda_{mk} = \frac{1}{(\mathbf{H}^T\mathbf{H})_{kk}} \text{ and } \mu_{nk} = \frac{1}{(\mathbf{W}^T\mathbf{W})_{kk}}. \tag{29}$$

The step-lengths chosen in the multiplicative updating rule is conservative enough so that the result is always nonnegative. On the other hand, the step-lengths chosen in the HALS/RRI algorithm could potentially lead to a nonnegative value, and therefore the projection $[\cdot]_+$ is needed. Although the convergence property of the BCD framework holds for the HALS/RRI algorithm as in Corollary 3, it does not hold for the multiplicative updating rule since the step-lengths in (28) does not achieve the optimal solution. In practice, the convergence of the HALS/RRI algorithm is much faster than that of the multiplicative updating.

Lee and Seung [67] showed that under the multiplicative updating rule, the objective function in (2) is non-increasing. However, it is unknown whether it converges to a stationary point. Gonzalez and Zhang demonstrated the difficulty [40], and the slow convergence of multiplicative updates has been further reported in [53,58,59,71]. To overcome this issue, Lin [70] proposed a modified update rule for which every limit point is stationary; note that, after this modification, the update rule becomes additive instead of multiplicative.

Since the values are updated only though multiplications, the elements of $\mathbf{W}$ and $\mathbf{H}$ obtained by the multiplicative updating rule typically remain nonzero. Hence, its solution matrices typically are denser than those from the BCD methods. The multiplicative updating rule breaks down if a zero value occurs to an element of the denominators in (27). To curcumvent this difficulty, practical implementations often add a small number, such as 10^{-16}, to each element of the denominators.

3.3 Alternating least squares method

In the two-block BCD method of Sect. 2.1, it is required to find a minimum of the nonnegativity-constrained least squares (NLS) subproblems in (7). Earlier, Berry et al. has proposed to approximately solve the NLS subproblems hoping to accelerate the algorithm [4]. In their alternating least squares (ALS) method, they solved the least squares problems ignoring the nonnegativity constraints, and then negative elements in the computed solution matrix are set to zeros. That is, $\mathbf{W}$ and $\mathbf{H}$ are updated as

$$\mathbf{W}^T \leftarrow \left[\left(\mathbf{H}^T\mathbf{H}\right)^{-1}\left(\mathbf{H}^T\mathbf{A}^T\right)\right]_+ \text{ and} \tag{30a}$$

$$\mathbf{H}^T \leftarrow \left[\left(\mathbf{W}^T\mathbf{W}\right)^{-1}\left(\mathbf{W}^T\mathbf{A}\right)\right]_+ . \tag{30b}$$

When $\mathbf{H}^T\mathbf{H}$ or $\mathbf{W}^T\mathbf{W}$ is rank-deficient, the Moore-Penrose pseudo-inverse may be used instead of the inverse operator. Unfortunately, results from (30) are not the minimizers of subproblems (7). Although each subproblem of the ALS method can be solved efficiently, the convergence property in Corollary 1 is not applicable to the ALS method. In fact, the ALS method does not necessarily decrease the objective function after each iteration [59].

It is interesting to note that the HALS/RRI method does not have this difficulty although the same element-wise projection is used. In the HALS/RRI method, a subproblem in the form of

$$\min_{\mathbf{x}\geq 0} \left\| \mathbf{b}\mathbf{x}^T - \mathbf{C} \right\|_F^2 \tag{31}$$

with $\mathbf{b} \in \mathbb{R}^M$ and $\mathbf{C} \in \mathbb{R}^{M\times N}$ is solved with $\mathbf{x} \leftarrow \left[\frac{\mathbf{C}^T\mathbf{b}}{\mathbf{b}^T\mathbf{b}}\right]_+$, which is the optimal solution of (31) as shown in Theorem 2. On the other hand, in the ALS algorithm, a subproblem in the form of

$$\min_{\mathbf{x}\geq 0} \|\mathbf{B}\mathbf{x} - \mathbf{c}\|_2^2 \tag{32}$$

with $\mathbf{B} \in \mathbb{R}^{M\times N}$ and $\mathbf{c} \in \mathbb{R}^M$ is solved with $\mathbf{x} \leftarrow \left[\left(\mathbf{B}^T\mathbf{B}\right)^{-1}\mathbf{B}^T\mathbf{c}\right]_+$, which is not an optimal solution of (32).

3.4 Successive rank one deflation

Some algorithms have been designed to compute NMF based on successive rank-one deflation. This approach is motivated from the fact that the singular value decomposition (SVD) can be computed through successive rank-one deflation. When considered for NMF, however, the rank-one deflation method has a few issues as we summarize below.

Let us first recapitulate the deflation approach for SVD. Consider a matrix $\mathbf{A} \in \mathbb{R}^{M\times N}$ of rank R, and suppose its SVD is written as

$$\mathbf{A} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^T = \sum_{r=1}^{R} \sigma_r \mathbf{u}_r \mathbf{v}_r^T, \tag{33}$$

where $\mathbf{U} = \left[\mathbf{u}_1 \ldots \mathbf{u}_R\right] \in \mathbb{R}^{M\times R}$ and $\mathbf{V} = \left[\mathbf{v}_1 \ldots \mathbf{v}_R\right] \in \mathbb{R}^{N\times R}$ are orthogonal matrices, and $\mathbf{\Sigma} \in \mathbb{R}^{R\times R}$ is a diagonal matrix having $\sigma_1 \geq \cdots \geq \sigma_R \geq 0$ in the diagonal. The rank-K SVD for $K < R$ is obtained by taking only the first K singular values and corresponding singular vectors:

$$\tilde{\mathbf{A}}_K = \tilde{\mathbf{U}}_K \tilde{\mathbf{\Sigma}}_K \tilde{\mathbf{V}}_K^T = \sum_{k=1}^{K} \sigma_k \mathbf{u}_k \mathbf{v}_k^T,$$

where $\tilde{\mathbf{U}}_K \in \mathbb{R}^{M\times K}$ and $\tilde{\mathbf{V}}_K \in \mathbb{R}^{N\times K}$ are sub-matrices of $\mathbf{U}$ and $\mathbf{V}$ obtained by taking the leftmost K columns. It is well-known that the best rank-K approximation of $\mathbf{A}$ in terms of minimizing the l_2-norm or the Frobenius norm of the residual matrix is the rank-K SVD: See Theorem 2.5.3 in Page 72 of Golub and Van Loan [39]. The rank-K SVD can be computed through successive rank one deflation as follows. First, the best rank-one approximation, $\sigma_1 \mathbf{u}_1 \mathbf{v}_1^T$, is computed with an efficient algorithm such as the power iteration. Then, the residual matrix is obtained as $\tilde{\mathbf{E}}_1 = \mathbf{A} - \sigma_1 \mathbf{u}_1 \mathbf{v}_1^T = \sum_{r=2}^{R} \sigma_r \mathbf{u}_r \mathbf{v}_r^T$, and the rank of $\tilde{\mathbf{E}}_1$ is $R-1$. For the residual matrix $\tilde{\mathbf{E}}_1$, its best rank-one approximation, $\sigma_2 \mathbf{u}_2 \mathbf{v}_2^T$, is obtained, and the residual matrix $\tilde{\mathbf{E}}_2$, whose rank is $R-2$, can be found in the same manner: $\tilde{\mathbf{E}}_2 = \tilde{\mathbf{E}}_1 - \sigma_2 \mathbf{u}_2 \mathbf{v}_2^T = \sum_{r=3}^{R} \sigma_r \mathbf{u}_r \mathbf{v}_r^T$. Repeating this process for K times, one can obtain the rank-K SVD.

When it comes to NMF, a notable theoretical result about nonnegative matrices relates SVD and NMF when $K = 1$. The following theorem, which extends the Perron-Frobenius theorem [3,45], is shown in Chapter 2 of Berman and Plemmons [3].

Theorem 3 *For a nonnegative symmetric matrix $\mathbf{A} \in \mathbb{R}_+^{N\times N}$, the eigenvalue of $\mathbf{A}$ with the largest magnitude is nonnegative, and there exists a nonnegative eigenvector corresponding to the largest eigenvalue.*

A direct consequence of Theorem 3 is the nonnegativity of the best rank-one approximation.

Corollary 4 *(Nonnegativity of best rank-one approximation) For any nonnegative matrix $\mathbf{A} \in \mathbb{R}_+^{M\times N}$, the following minimization problem*

$$\min_{\mathbf{u}\in\mathbb{R}^M, \mathbf{v}\in\mathbb{R}^N} \left\| \mathbf{A} - \mathbf{u}\mathbf{v}^T \right\|_F^2. \tag{34}$$

has an optimal solution satisfying $\mathbf{u} \geq 0$ and $\mathbf{v} \geq 0$.

Another way to realizing Corollary 4 is through the use of the SVD. For a nonnegative matrix $\mathbf{A} \in \mathbb{R}_+^{M\times N}$ and for any vectors $\mathbf{u} \in \mathbb{R}^M$ and $\mathbf{v} \in \mathbb{R}^N$,

$$\begin{aligned} \left\| \mathbf{A} - \mathbf{u}\mathbf{v}^T \right\|_F^2 &= \sum_{m=1}^{M} \sum_{n=1}^{N} (a_{mn} - u_m v_n)^2 \\ &\geq \sum_{m=1}^{M} \sum_{n=1}^{N} (a_{mn} - |u_m|\,|v_n|)^2 . \end{aligned} \tag{35}$$

Hence, element-wise absolute values can be taken from the left and right singular vectors that correspond to the largest singular value to achieve the best rank-one approximation satisfying nonnegativity. There might be other optimal solutions of (34) involving negative numbers: See [34].

The elegant property in Corollary 4, however, is not readily applicable when $K \geq 2$. After the best rank-one approximation matrix is deflated, the residual matrix may contain negative elements, and then Corollary 4 is not applicable any more. In general, successive rank-one deflation is not an optimal approach for NMF computation. Let us take a look at a small example which demonstrates this issue. Consider matrix $\mathbf{A}$ given as

$$\mathbf{A} = \begin{pmatrix} 4 & 6 & 0 \\ 6 & 4 & 0 \\ 0 & 0 & 1 \end{pmatrix}.$$

The best rank-one approximation of $\mathbf{A}$ is shown as $\mathbf{A}_1$ below. The residual is $\mathbf{E}_1 = \mathbf{A} - \mathbf{A}_1$, which contains negative elements:

$$\mathbf{A}_1 = \begin{pmatrix} 5 & 5 & 0 \\ 5 & 5 & 0 \\ 0 & 0 & 0 \end{pmatrix}, \; \mathbf{E}_1 = \begin{pmatrix} -1 & 1 & 0 \\ 1 & -1 & 0 \\ 0 & 0 & 1 \end{pmatrix}.$$

One of the best rank-one approximations of $\mathbf{E}_1$ with nonnegativity constraints is $\mathbf{A}_2$, and the residual matrix is $\mathbf{E}_2 = \mathbf{A}_1 - \mathbf{A}_2$:

$$\mathbf{A}_2 = \begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 1 \end{pmatrix}, \; \mathbf{E}_2 = \begin{pmatrix} -1 & 1 & 0 \\ 1 & -1 & 0 \\ 0 & 0 & 0 \end{pmatrix}.$$

The nonnegative rank-two approximation obtained by this rank-one deflation approach is

$$\mathbf{A}_1 + \mathbf{A}_2 = \begin{pmatrix} 5 & 5 & 0 \\ 5 & 5 & 0 \\ 0 & 0 & 1 \end{pmatrix}.$$

However, the best nonnegative rank-two approximation of $\mathbf{A}$ is in fact $\hat{\mathbf{A}}_2$ with residual matrix $\hat{\mathbf{E}}_2$:

$$\hat{\mathbf{A}}_2 = \begin{pmatrix} 4 & 6 & 0 \\ 6 & 4 & 0 \\ 0 & 0 & 0 \end{pmatrix}, \; \hat{\mathbf{E}}_2 = \begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 1 \end{pmatrix}.$$

Therefore, a strategy that successively finds the best rank-one approximation with nonnegativity constraints and deflates in each step does not necessarily lead to an optimal solution of NMF.

Due to this difficulty, some variations of rank-one deflation have been investigated for NMF. Biggs et al. [6] proposed a rank-one reduction algorithm in which they look for a nonnegative submatrix that is close to a rank-one approximation. Once such a submatrix is identified, they compute the best rank-one approximation using the power method and ignore the residual. Gillis and Glineur [36] sought a nonnegative rank-one approximation under the constraint that the residual matrix remains element-wise nonnegative. Due to the constraints, however, the problem of finding the nonnegative rank-one approximation becomes more complicated and computationally expensive than the power iteration. Optimization properties such as a convergence to a stationary point has not been shown for these modified rank-one reduction methods.

It is worth noting the difference between the HALS/RRI algorithm, described as the $2K$ vector block case in Sect. 2.2, and the rank-one deflation method. These approaches are similar in that the rank-one approximation problem with nonnegativity constraints is solved in each step, filling in the kth columns of $\mathbf{W}$ and $\mathbf{H}$ with the solution for $k = 1, \ldots, K$. In the rank-one deflation method, once the kth columns of $\mathbf{W}$ and $\mathbf{H}$ are computed, they are fixed and kept as a part of the final solution before the $(k+1)$th columns are computed. On the other hand, the HALS/RRI algorithm updates all the columns through multiple iterations until a local minimum is achieved. This simultaneous searching for all $2K$ vectors throughout the iterations is necessary to achieve an optimal solution of NMF, unlike in the case of SVD.

4 Algorithms for the nonnegativity constrained least squares problems

We review numerical methods developed for the NLS subproblems in (7). For simplicity, we consider the following notations in this section:

$$\min_{\mathbf{X}\geq 0} \|\mathbf{BX}-\mathbf{C}\|_F^2 = \sum_{r=1}^{R} \|\mathbf{B}\mathbf{x}_r - \mathbf{c}_r\|_2^2\,, \tag{36}$$

where $\mathbf{B} \in \mathbb{R}^{P\times Q}$, $\mathbf{C} = [\mathbf{c}_1, \ldots, \mathbf{c}_R] \in \mathbb{R}^{Q\times R}$, and $\mathbf{X} = [\mathbf{x}_1, \ldots, \mathbf{x}_R] \in \mathbb{R}^{Q\times R}$. We mainly discuss two groups of algorithms for the NLS problems. The first group consists of the gradient descent and the Newton-type methods that are modified to satisfy the nonnegativity constraints using a projection operator. The second group consists of the active-set and the active-set-like methods, in which zero and nonzero variables are explicitly kept track of and a system of linear equations is solved at each iteration. For more details, see Lawson and Hanson [64], Bjork [8], and Chen and Plemmons [15].

To facilitate our discussion, we state a simple NLS problem with a single right-hand side:

$$\min_{\mathbf{x}\geq 0} g(\mathbf{x}) = \|\mathbf{Bx}-\mathbf{c}\|_2^2. \tag{37}$$

Problem (36) may be solved by handling independent problems for the columns of $\mathbf{X}$, whose form appears as (37). Otherwise, the problem in (36) can also be transformed into

$$\min_{\mathbf{x}_1,\ldots,\mathbf{x}_R\geq 0} \left\| \begin{pmatrix} \mathbf{B} & & \\ & \ddots & \\ & & \mathbf{B} \end{pmatrix} \begin{pmatrix} \mathbf{x}_1 \\ \vdots \\ \mathbf{x}_R \end{pmatrix} - \begin{pmatrix} \mathbf{c}_1 \\ \vdots \\ \mathbf{c}_R \end{pmatrix} \right\|_2^2. \tag{38}$$

4.1 Projected iterative methods

Projected iterative methods for the NLS problems are designed based on the fact that the objective function in (36) is differentiable and that the projection to the nonnegative orthant is easy to compute. The first method of this type proposed for NMF was the projected gradient method of Lin [71]. Their update formula is written as

$$\mathbf{x}^{(i+1)} \leftarrow \left[\mathbf{x}^{(i)} - \alpha^{(i)}\nabla g(\mathbf{x}^{(i)})\right]_+, \tag{39}$$

where $\mathbf{x}^{(i)}$ and $\alpha^{(i)}$ represent the variables and the step length at the ith iteration. Step length $\alpha^{(i)}$ is chosen by a back-tracking line search to satisfy Armijo's rule with an optional stage that increases the step length. Kim et al. [51] proposed a quasi-Newton method by utilizing the second order information to improve convergence:

$$\mathbf{x}^{(i+1)} \leftarrow \begin{pmatrix} \left[\mathbf{y}^{(i)} - \alpha^{(i)}\mathbf{D}^{(i)}\nabla g(\mathbf{y}^{(i)})\right]_+ \\ \mathbf{0} \end{pmatrix}, \tag{40}$$

where $\mathbf{y}^{(i)}$ is a subvector of $\mathbf{x}^{(i)}$ with elements that are not optimal in terms of the Karush–Kuhn–Tucker (KKT) conditions. They efficiently updated $\mathbf{D}^{(i)}$ using the BFGS method and selected $\alpha^{(i)}$ by a back-tracking line search. Whereas Lin considered a stacked-up problem as in (38), the quasi-Newton method by Kim et al. was applied to each column separately.

A notable variant of the projected gradient method is the Barzilai-Borwein method [7]. Han et al. [42] proposed alternating projected Barzilai-Borwein method for NMF. A key characteristic of the Barzilai-Borwein method in unconstrained quadratic programming is that the step-length is chosen by a closed-form formula without having to perform a line search:

$$\mathbf{x}^{(i+1)} \leftarrow \left[\mathbf{x}^{(i)} - \alpha^{(i)}\nabla g(\mathbf{x}^{(i)})\right]_+ \text{ with } \alpha^{(i)} = \frac{\mathbf{s}^T\mathbf{s}}{\mathbf{y}^T\mathbf{s}}, \text{ where}$$
$$\mathbf{s} = \mathbf{x}^{(i)} - \mathbf{x}^{(i-1)} \text{ and } \mathbf{y} = \nabla g(\mathbf{x}^{(i)}) - \nabla g(\mathbf{x}^{(i-1)}).$$

Algorithm 1 Outline of the Active-set Method for $\min_{\mathbf{x}\geq 0} g(\mathbf{x}) = \|\mathbf{B}\mathbf{x} - \mathbf{c}\|_2^2$ (See [64] for more details)

```
1: Initialize $\mathbf{x}$ (typically with all zeros).
2: Set $\mathcal{I}, \mathcal{E}$ (working sets) to be indices representing zero and nonzero variables. Let $\mathbf{x}_\mathcal{I}$ and $\mathbf{x}_\mathcal{E}$ denote the subvectors of $\mathbf{x}$ with corresponding indices, and let $\mathbf{B}_\mathcal{I}$ and $\mathbf{B}_\mathcal{E}$ denote the submatrices of $\mathbf{B}$ with corresponding column indices.
3: for $i = 1, 2, \ldots$ do
4:     Solve an unconstrained least squares problem,
           $$\min_{\mathbf{z}} \left\|\mathbf{B}_\mathcal{E}\mathbf{z} - \mathbf{c}\right\|_2^2. \quad (41)$$
5:     Check if the solution is nonegative and satisfies KKT conditions. If so, set $\mathbf{x}_\mathcal{E} \leftarrow \mathbf{z}$, set $\mathbf{x}_\mathcal{I}$ with zeros, and return $\mathbf{x}$ as a solution. Otherwise, update $\mathbf{x}, \mathcal{I}$, and $\mathcal{E}$.
6: end for
```

When the nonnegativity constraints are given, however, back-tracking line search still had to be employed. Han et al. discussed a few variations of the Barzilai-Borwein method for NMF and reported that the algorithms outperform Lin's method.

Many other methods have been developed. Merritt and Zhang [73] proposed an interior point gradient method, and Friedlander and Hatz [32] used a two-metric projected gradient method in their study on NTF. Zdunek and Cichocki [86] proposed a quasi-Newton method, but its lack of convergence was pointed out [51]. Zdunek and Cichocki [87] also studied the projected Landweber method and the projected sequential subspace method.

4.2 Active-set and active-set-like methods

The active-set method for the NLS problems is due to Lawson and Hanson [64]. A key observation is that, if the zero and nonzero elements of the final solution are known in advance, the solution can be easily computed by solving an unconstrained least squares problem for the nonzero variables and setting the rest to zeros. The sets of zero and nonzero variables are referred to as *active* and *passive* sets, respectively. In the active-set method, so-called *workings sets* are kept track of until the optimal active and passive sets are found. A rough pseudo-code for the active-set method is shown in Algorithm 1.

Lawson and Hanson's method has been a standard for the NLS problems, but applying it directly to NMF is very slow. When used for NMF, it can be accelerated in two different ways. The first approach is to use the QR decomposition to solve (41) or the Cholesky decomposition to solve the normal equations $\left(\mathbf{B}_\mathcal{E}^T\mathbf{B}_\mathcal{E}\right)\mathbf{z} = \mathbf{B}_\mathcal{E}^T\mathbf{c}$ and have the Cholesky or QR factors updated by the Givens rotations [39]. The second approach, which was proposed by Bro and De Jong [9] and Ven Benthem and Keenan [81], is to identify common computations in solving the NLS problems with multiple right-hand sides. More information and experimental comparisons of these two approaches are provided in [59].

The active-set methods possess a property that the objective function decreases after each iteration; however, maintaining this property often limits its scalability. A main computational burden of the active-set methods is in solving the unconstrained least squares problem (41); hence, the number of iterations required until termination considerably affects the computation cost. In order to achieve the monotonic decreasing property, typically only one variable is exchanged between working sets per iteration. As a result, when the number of unknowns is large, the number of iterations required for termination grows, slowing down the method. The block principal pivoting method developed by Kim and Park [58,59] overcomes this limitation. Their method, which is based on the work of Judice and Pires [50], allows the exchanges of multiple variables between working sets. This method does not maintain the nonnegativity of intermediate vectors nor the monotonic decrease of the objective function, but it requires a smaller number of iterations until termination than the active set methods. It is worth emphasizing that the grouping-based speed-up technique, which was earlier devised for the active-set method, is also effective with the block principal pivoting method for the NMF computation: For more details, see [59].

4.3 Discussion and other methods

A main difference between the projected iterative methods and the active-set-like methods for the NLS problems lies in their convergence or termination. In projected iterative methods, a sequence of tentative solutions is generated so that an optimal solution is approached in the limit. In practice, one has to somehow stop iterations and return the current estimate, which might be only an approximation of the solution. In the active-set and active-set-like methods, in contrast, there is no concept of a limit point. Tentative solutions are generated with a goal of finding the optimal active and passive set partitioning, which is guaranteed to be found in a finite number of iterations since there are only a finite number of possible active and passive set partitionings. Once the optimal active and passive sets are found, the methods terminate. There are trade-offs of these behavior. While the projected iterative methods may return an approximate solution after a few number of iterations, the active-set and active-set-like methods only return a solution after they terminate. After the termination, however, the solution from the active-set-like methods is an exact solution only subject to numerical rounding errors while the solution from the projected iterative methods might be an approximate one.

Other approaches for solving the NLS problems can be considered as a subroutine for the NMF computation. Bellavia et al. [2] have studied an interior point Newton-like method, and Franc et al. [31] presented a sequential coordinate-wise method. Some observations about the NMF computation based on these methods as well as other methods are offered in Cichocki et al. [22]. Chu and Lin [18] proposed an algorithm based on low-dimensional polytope approximation: Their algorithm is motivated by a geometrical interpretation of NMF that data points are approximated by a simplicial cone [27].

Different conditions are required for the NLS algorithms to guarantee convergence or termination. The requirement of the projected gradient method [71] is mild as it only requires an appropriate selection of the step-length. Both the quasi-Newton method [51] and the interior point gradient method [73] require that matrix $\mathbf{B}$ in (37) is of full column rank. The active-set method [53,64] does not require the full-rank condition as long as a zero vector is used for initialization [28]. In the block principal pivoting method [58,59], on the other hand, the full-rank condition is required. Since NMF is formulated as a lower rank approximation and K is typically much smaller than the rank of the input matrix, the ranks of both $\mathbf{W}$ and $\mathbf{H}$ in (7) typically remain full. When this condition is not likely to be satisfied, the Frobenius-norm regularization of Sect. 2.4 can be adopted to guarantee the full rank condition.

5 BCD framework for nonnegative CP

Our discussion on the low-rank factorizations of nonnegative matrices naturally extends to those of nonnegative tensors. In this section, we discuss nonnegative CANDECOMP/PARAFAC (NCP) and explain how it can be computed by the BCD framework.

A few other decomposition models of higher order tensors have been studied, and interested readers are referred to [1,61]. The organization of this section is similar to that of Sect. 2, and we will show that the NLS algorithms reviewed in Sect. 4 can also be used to factorize tensors.

Let us consider an Nth-order tensor $\mathcal{A} \in \mathbb{R}^{M_1 \times \cdots \times M_N}$. For an integer K, we are interested in finding nonnegative factor matrices $\mathbf{H}^{(1)}, \ldots, \mathbf{H}^{(N)}$ where $\mathbf{H}^{(n)} \in \mathbb{R}^{M_n \times K}$ for $n = 1, \ldots, N$ such that

$$\mathcal{A} \approx [\![\mathbf{H}^{(1)}, \ldots, \mathbf{H}^{(N)}]\!], \tag{42}$$

where

$$\mathbf{H}^{(n)} = \left[\mathbf{h}_1^{(n)} \ \ldots \ \mathbf{h}_K^{(n)} \right] \ \text{for}\, n = 1, \ldots, N \ \text{and} \tag{43}$$

$$[\![\mathbf{H}^{(1)}, \ldots, \mathbf{H}^{(N)}]\!] = \sum_{k=1}^{K} \mathbf{h}_k^{(1)} \circ \cdots \circ \mathbf{h}_k^{(N)}. \tag{44}$$

The '$\circ$' symbol represents the outer product of vectors, and a tensor in the form of $\mathbf{h}_k^{(1)} \circ \cdots \circ \mathbf{h}_k^{(N)}$ is called a *rank-one* tensor. Model (42) is known as CANDECOMP/PARAFAC (CP) [14,43]: In the CP decomposition, $\mathcal{A}$ is represented as the sum of K rank-one tensors. The smallest integer K for which (42) holds as equality is called the *rank* of tensor $\mathcal{A}$. The CP decomposition reduces to a matrix decomposition if $N = 2$. The nonnegative CP decomposition is obtained by adding nonnegativity constraints to factor matrices $\mathbf{H}^{(1)}, \ldots, \mathbf{H}^{(N)}$. A corresponding problem can be written as, for $\mathcal{A} \in \mathbb{R}_+^{M_1 \times \cdots \times M_N}$,

$$\min_{\mathbf{H}^{(1)}, \ldots, \mathbf{H}^{(N)}} f(\mathbf{H}^{(1)}, \ldots, \mathbf{H}^{(N)}) = \left\| \mathcal{A} - [\![\mathbf{H}^{(1)}, \ldots, \mathbf{H}^{(N)}]\!] \right\|_F^2,$$
$$\text{s.t. } \mathbf{H}^{(n)} \geq 0 \text{ for } n = 1, \ldots N. \tag{45}$$

We discuss algorithms for solving (45) in this section [19,32,54,60]. Toward that end, we introduce definitions of some operations of tensors.

Mode-n matricization: The mode-n matricization of $\mathcal{A} \in \mathbb{R}^{M_1 \times \cdots \times M_N}$, denoted by $\mathbf{A}^{<n>}$, is a matrix obtained by linearizing all the indices of tensor $\mathcal{A}$ except n. Specifically, $\mathbf{A}^{<n>}$ is a matrix of size $M_n \times (\prod_{\tilde{n}=1, \tilde{n} \neq n}^{N} M_{\tilde{n}})$, and the $(m_1, \ldots, m_N)$th element of $\mathcal{A}$ is mapped to the (m_n, J)th element of $\mathbf{A}^{<n>}$ where

$$J = 1 + \sum_{j=1}^{N} (m_j - 1) J_j \ \text{and} \ J_j = \prod_{l=1, l \neq n}^{j-1} M_l.$$

Mode-n fibers: The fibers of higher-order tensors are vectors obtained by specifying all indices except one. Given a tensor $\mathcal{A} \in \mathbb{R}^{M_1 \times \cdots \times M_N}$, a mode-n fiber denoted by $\mathbf{a}_{m_1 \ldots m_{n-1} : m_{n+1} \ldots m_N}$ is a vector of length M_n with all the elements having $m_1, \ldots, m_{n-1}$, $m_{n+1}, \ldots, m_N$ as indices for the 1st, $\ldots$, $(n-1)$th, $(n+2)$th, $\ldots$, Nth orders. The columns and the rows of a matrix are the mode-1 and the mode-2 fibers, respectively.

Mode-n product: The mode-n product of a tensor $\mathcal{A} \in \mathbb{R}^{M_1 \times \cdots \times M_N}$ and a matrix $\mathbf{U} \in \mathbb{R}^{J \times M_n}$, denoted by $\mathcal{A} \times_n \mathbf{U}$, is a tensor obtained by multiplying all mode-n fibers of $\mathcal{A}$ with the columns of $\mathbf{U}$. The result is a tensor of size $M_1 \times \cdots \times M_{n-1} \times J \times M_{n+1} \times \cdots \times M_N$ having elements as

$$(\mathcal{A} \times_n \mathbf{U})_{m_1 \ldots m_{n-1} j m_{n+1} \ldots m_N} = \sum_{m_n=1}^{M_n} x_{m_1 \ldots m_N} u_{j m_n}.$$

In particular, the mode-n product of $\mathcal{A}$ and a vector $\mathbf{u} \in \mathbb{R}^{M_n}$ is a tensor of size $M_1 \times \cdots \times M_{n-1} \times M_{n+1} \times \cdots \times M_N$.
Khatri-Rao product: The Khatri-Rao product of two matrices $\mathbf{A} \in \mathbb{R}^{J_1 \times L}$ and $\mathbf{B} \in \mathbb{R}^{J_2 \times L}$, denoted by $\mathbf{A} \odot \mathbf{B} \in \mathbb{R}^{(J_1 J_2) \times L}$, is defined as

$$\mathbf{A} \odot \mathbf{B} = \begin{bmatrix} a_{11}\mathbf{b}_1 & a_{12}\mathbf{b}_2 & \cdots & a_{1L}\mathbf{b}_L \\ a_{21}\mathbf{b}_1 & a_{22}\mathbf{b}_2 & \cdots & a_{2L}\mathbf{b}_L \\ \vdots & \vdots & \ddots & \vdots \\ a_{J_1 1}\mathbf{b}_1 & a_{J_1 2}\mathbf{b}_2 & \cdots & a_{J_1 L}\mathbf{b}_L \end{bmatrix}.$$

5.1 BCD with N matrix blocks

A simple BCD method can be designed for (45) considering each of the factor matrics $\mathbf{H}^{(1)}, \ldots, \mathbf{H}^{(N)}$ as a block. Using notations introduced above, approximation model (42) can be written as, for any $n \in \{1, \ldots, N\}$,

$$\mathbf{A}^{<n>} \approx \mathbf{H}^{(n)} \left(\mathbf{B}^{(n)}\right)^T, \tag{46}$$

where

$$\mathbf{B}^{(n)} = \mathbf{H}^{(N)} \odot \cdots \odot \mathbf{H}^{(n+1)} \odot \mathbf{H}^{(n-1)} \odot \cdots \odot \mathbf{H}^{(1)}$$

$$\in \mathbb{R}^{\left(\prod_{\tilde{n}=1,\tilde{n}\neq n}^{N} M_{\tilde{n}}\right) \times K}. \tag{47}$$

Representation in (46) simplifies the treatment of this N matrix block case. After $\mathbf{H}^{(2)}, \ldots, \mathbf{H}^{(N)}$ are initialized with nonnegative elements, the following subproblem is solved iteratively for $n = 1, \ldots N$:

$$\mathbf{H}^{(n)} \leftarrow \underset{\mathbf{H} \geq 0}{\arg\min} \left\| \mathbf{B}^{(n)} \mathbf{H}^T - \left(\mathbf{A}^{<n>}\right)^T \right\|_F^2. \tag{48}$$

Since subproblem (48) is an NLS problem, as in the matrix factorization case, this matrix-block BCD method is called the alternating nonnegative least squares (ANLS) framework [32,54,60]. The convergence property of the BCD method in Theorem 1 yields the following corollary.

Corollary 5 *If a unique solution exists for* (48) *and is attained for* $n = 1, \ldots, N$*, then every limit point of the sequence* $\left\{\left(\mathbf{H}^{(1)}, \ldots, \mathbf{H}^{(N)}\right)^{(i)}\right\}$ *generated by the ANLS framework is a stationary point of* (45).

In particular, if each $\mathbf{B}^{(n)}$ is of full column rank, the subproblem has a unique solution. Algorithms for the NLS subproblems discussed in Sect. 4 can be used to solve (48).

For higher order tensors, the number of rows in $\mathbf{B}^{(n)}$ and $(\mathbf{A}^{<n>})^T$, i.e., $\prod_{\tilde{n}=1,\tilde{n}\neq n}^{N} M_{\tilde{n}}$, can be quite large. However, often $\mathbf{B}^{(n)}$ and $(\mathbf{A}^{<n>})^T$ do not need to be explicitly constructed. In most algorithms explained in Sect. 4, it is enough to have $\mathbf{B}^{(n)T} \left(\mathbf{A}^{<n>}\right)^T$ and $\left(\mathbf{B}^{(n)}\right)^T \mathbf{B}^{(n)}$. It is easy to verify that $\mathbf{B}^{(n)T} \left(\mathbf{A}^{<n>}\right)^T$ can be obtained by successive mode-n products:

$$\mathbf{B}^{(n)T} \left(\mathbf{A}^{<n>}\right)^T = \mathcal{A} \times_1 \mathbf{H}^{(1)} \ldots \times_{(n-1)} \mathbf{H}^{(n-1)}$$

$$\times_{(n)} \mathbf{H}^{(n)} \ldots \times_{(N)} \mathbf{H}^{(N)}. \tag{49}$$

In addition, $\left(\mathbf{B}^{(n)}\right)^T \mathbf{B}^{(n)}$ can be obtained as

$$\left(\mathbf{B}^{(n)}\right)^T \mathbf{B}^{(n)} = \bigotimes_{\tilde{n}=1,\tilde{n}\neq n}^{N} \left(\mathbf{H}^{(\tilde{n})}\right)^T \mathbf{H}^{(\tilde{n})}, \tag{50}$$

where $\bigotimes$ represents element-wise multiplication.

5.2 BCD with KN vector blocks

Another way to apply the BCD framework to (45) is to treat each column of $\mathbf{H}^{(1)}, \ldots, \mathbf{H}^{(N)}$ as a block. The columns are updated by solving, for $n = 1, \ldots N$ and for $k = 1, \ldots, K$,

$$\mathbf{h}_k^{(n)} \leftarrow$$

$$\underset{\mathbf{h}\geq 0}{\arg\min} \left\| [\![\mathbf{h}_k^{(1)}, \ldots, \mathbf{h}_k^{(n-1)}, \mathbf{h}, \mathbf{h}_k^{(n+1)}, \cdots, \mathbf{h}_k^{(N)}]\!] - \mathcal{R}_k \right\|_F^2. \tag{51}$$

where

$$\mathcal{R}_k = \mathcal{A} - \sum_{\tilde{k}=1,\tilde{k}\neq k}^{K} \mathbf{h}_{\tilde{k}}^{(1)} \circ \cdots \circ \mathbf{h}_{\tilde{k}}^{(N)}.$$

Using matrix notations, problem (51) can be rewritten as

$$\mathbf{h}_k^{(n)} \leftarrow \underset{\mathbf{h}\geq 0}{\arg\min} \left\| \mathbf{b}_k^{(n)} \mathbf{h}^T - \left(\mathbf{R}_k^{<n>}\right)^T \right\|_F^2, \tag{52}$$

where $\mathbf{R}_k^{<n>}$ is the mode-n matricization of $\mathcal{R}_k$ and

$$\mathbf{b}_k^{(n)} = \mathbf{h}_k^{(N)} \odot \cdots \odot \mathbf{h}_k^{(n+1)} \odot \mathbf{h}_k^{(n-1)} \odot \cdots \odot \mathbf{h}_k^{(1)}$$

$$\in \mathbb{R}^{\left(\prod_{\tilde{n}=1,\tilde{n}\neq n}^{N} M_{\tilde{n}}\right)\times 1}. \tag{53}$$

This vector-block BCD method corresponds to the HALS method by Cichocki et al. for NTF [19,22]. The convergence property in Theorem 1 yields the following corollary.

Corollary 6 *If a unique solution exists for* (52) *and is attained for* $n = 1, \ldots, N$ *and for* $k = 1, \ldots, K$, *every limit point of the sequence* $\left\{\left(\mathbf{H}^{(1)}, \ldots, \mathbf{H}^{(N)}\right)^{(i)}\right\}$ *generated by the vector-block BCD method is a stationary point of* (45).

Using Theorem 2, the solution of (52) is

$$\mathbf{h}_k^{(n)} \leftarrow \frac{\left[\mathbf{R}_k^{<n>} \mathbf{b}_k^{(n)}\right]_+}{\left\|\mathbf{b}_k^{(n)}\right\|_2^2}. \tag{54}$$

Solution (54) can be evaluated without constructing $\mathbf{R}_k^{<n>}$. Observe that

$$\left(\mathbf{b}_k^{(n)}\right)^T \mathbf{b}_k^{(n)} = \prod_{\tilde{n}=1,\tilde{n}\neq n}^{N} \left(\mathbf{h}_k^{(\tilde{n})}\right)^T \mathbf{h}_k^{(\tilde{n})}, \tag{55}$$

which is a simple case of (50), and

$$\mathbf{R}_k^{<n>}\mathbf{b}_k^{(n)} = \left(\mathcal{A} - \sum_{\tilde{k}=1,\tilde{k}\neq k}^{K} \mathbf{h}_{\tilde{k}}^{(1)} \circ \cdots \circ \mathbf{h}_{\tilde{k}}^{(N)}\right)^{<n>} \mathbf{b}_k^{(n)} \tag{56}$$

$$= \mathbf{A}^{<n>}\mathbf{b}_k^{(n)} - \sum_{\tilde{k}=1,\tilde{k}\neq k}^{K} \left(\prod_{\tilde{n}=1,\tilde{n}\neq n}^{N} \left(\mathbf{h}_{\tilde{k}}^{(\tilde{n})}\right)^T \mathbf{h}_k^{(\tilde{n})}\right) \mathbf{h}_{\tilde{k}}^{(n)}. \tag{57}$$

Solution (54) can then be simplified as

$$\mathbf{h}_k^{(n)} \leftarrow \left[\mathbf{h}_k^{(n)} + \frac{\mathbf{A}^{<n>}\mathbf{b}_k^{(n)} - \mathbf{H}^{(n)}\left(\bigotimes_{\tilde{n}=1,\tilde{n}\neq n}^{N}\left(\mathbf{H}^{(\tilde{n})}\right)^T\mathbf{H}^{(\tilde{n})}\right)_{\cdot k}}{\prod_{\tilde{n}=1,\tilde{n}\neq n}^{N}\left(\mathbf{h}_k^{(\tilde{n})}\right)^T \mathbf{h}_k^{(\tilde{n})}}\right]_+, \tag{58}$$

where $\mathbf{A}^{<n>}\mathbf{b}_k^{(n)}$ can be computed using (49). Observe the similarity between (58) and (14).

6 Implementation issues and comparisons

6.1 Stopping criterion

Iterative methods have to be equipped with a criterion for stopping iterations. In NMF or NTF, an ideal criterion would be to stop iterations after a local minimum of (2) or (45) is attained. In practice, a few alternatives have been used.

Let us first discuss stopping criteria for NMF. A naive approach is to stop when the decrease of the objective function becomes smaller than some predefined threshold:

$$f(\mathbf{W}^{(i-1)}, \mathbf{H}^{(i-1)}) - f(\mathbf{W}^{(i)}, \mathbf{H}^{(i)}) \leq \epsilon, \tag{59}$$

where ϵ is a tolerance value to choose. Although this method is commonly adopted, it is potentially misleading because the decrease of the objective function may become small before a local minimum is achieved. A more principled criterion was proposed by Lin as follows [71]. According to the Karush–Kuhn–Tucher (KKT) conditions, $(\mathbf{W}, \mathbf{H})$ is a stationary point of (2) if and only if [17]

$$\mathbf{W} \geq 0, \quad \mathbf{H} \geq 0, \tag{60a}$$

$$\nabla f_{\mathbf{W}} = \frac{\partial f(\mathbf{W}, \mathbf{H})}{\partial \mathbf{W}} \geq 0, \quad \nabla f_{\mathbf{H}} = \frac{\partial f(\mathbf{W}, \mathbf{H})}{\partial \mathbf{H}} \geq 0, \tag{60b}$$

$$\mathbf{W} \bigotimes \nabla f_{\mathbf{W}} = 0, \quad \mathbf{H} \bigotimes \nabla f_{\mathbf{H}} = 0, \tag{60c}$$

where

$$\nabla f_{\mathbf{W}} = 2\mathbf{W}\mathbf{H}^T\mathbf{H} - 2\mathbf{A}\mathbf{H} \text{ and}$$
$$\nabla f_{\mathbf{H}} = 2\mathbf{H}\mathbf{W}^T\mathbf{W} - 2\mathbf{A}^T\mathbf{W}.$$

Define the projected gradient $\nabla^p f_{\mathbf{W}} \in \mathbb{R}^{M \times K}$ as,

$$\left(\nabla^p f_{\mathbf{W}}\right)_{mk} \equiv \begin{cases} (\nabla f_{\mathbf{W}})_{mk} & \text{if } (\nabla f_{\mathbf{W}})_{mk} < 0 \text{ or } \mathbf{W}_{mk} > 0, \\ 0 & \text{otherwise,} \end{cases}$$

for $m = 1, \ldots, M$ and $k = 1, \ldots, K$, and $\nabla^p f_{\mathbf{H}}$ similarly. Then, conditions (60) can be rephrased as

$$\nabla^p f_{\mathbf{W}} = \mathbf{0} \ \text{ and } \ \nabla^p f_{\mathbf{H}} = \mathbf{0}.$$

Denote the projected gradient matrices at the ith iteration by $\nabla^p f_{\mathbf{W}}^{(i)}$ and $\nabla^p f_{\mathbf{H}}^{(i)}$, and define

$$\Delta(i) = \sqrt{\left\|\nabla^p f_{\mathbf{W}}^{(i)}\right\|_F^2 + \left\|\nabla^p f_{\mathbf{H}}^{(i)}\right\|_F^2}. \tag{61}$$

Using this definition, the stopping criterion is written by

$$\frac{\Delta(i)}{\Delta(0)} \le \epsilon, \tag{62}$$

where $\Delta(0)$ is from the initial values of $(\mathbf{W}, \mathbf{H})$. Unlike (59), (62) guarantees the stationarity of the final solution. Variants of this criterion was used in [40,53].

An analogous stopping criterion can be derived for the NCP formulation in (45). The gradient matrix $\nabla f_{\mathbf{H}^{(n)}}$ can be derived from the least squares representation in (48):

$$\nabla f_{\mathbf{H}^{(n)}} = 2\mathbf{H}^{(n)} \left(\mathbf{B}^{(n)}\right)^T \mathbf{B}^{(n)} - 2\mathbf{A}^{<n>}\mathbf{B}^{(n)}.$$

See (49) and (50) for efficient computation of $\left(\mathbf{B}^{(n)}\right)^T \mathbf{B}^{(n)}$ and $\mathbf{A}^{<n>}\mathbf{B}^{(n)}$. With function Δ defined as

$$\Delta(i) = \sqrt{\sum_{n=1}^{N} \left\|\nabla^p f_{\mathbf{H}^{(n)}}^{(i)}\right\|_F^2}, \tag{63}$$

criterion in (62) can be used to stop iterations of an NCP algorithm.

Using (59) or (62) for the purpose of comparing algorithms might be unreliable. One might want to measure the amounts of time until several algorithms satisfy one of these criteria and compare them [53,71]. Such comparison would usually reveal meaningful trends, but there are some caveats. The difficulty of using (59) is straightforward because, in some algorithm such as the multiplicative updating rule, the difference in (59) can become quite small before arriving at a minimum. The difficulty of using (62) is as follows. Note that the diagonal rescaling of $\mathbf{W}$ and $\mathbf{H}$ does not affect the quality of approximation: For a diagonal matrix $\mathbf{D} \in \mathbb{R}_+^{K \times K}$ with positive diagonal elements, $\mathbf{W}\mathbf{H}^T = \mathbf{W}\mathbf{D}^{-1} (\mathbf{H}\mathbf{D})^T$. However, the norm of the projected gradients in (61) is affected by a diagonal scaling:

$$\left(\frac{\partial f}{\partial(\mathbf{W}\mathbf{D}^{-1})}, \frac{\partial f}{\partial(\mathbf{H}\mathbf{D})}\right) = \left(\left(\frac{\partial f}{\partial \mathbf{W}}\right)\mathbf{D}, \left(\frac{\partial f}{\partial \mathbf{H}}\right)\mathbf{D}^{-1}\right).$$

Hence, two solutions that are only different up to a diagonal scaling have the same objective function value, but they can be measured differently in terms of the norm of the projected gradients. See Kim and Park [59] for more information. Ho [44] considered including a normalization step before computing (61) to avoid this issue.

6.2 Results of experimental comparisons

A number of papers have reported the results of experimental comparisons of NMF algorithms. A few papers have shown the slow convergence of Lee and Seung's multiplicative updating rule and demonstrated the superiority of other algorithms published subsequently [40,53,71]. Comprehensive comparisons of several efficient algorithms for NMF were conducted in Kim and Park [59], where MATLAB implementations of the ANLS-based methods, the HALS/RRI method, the multiplicative updating rule, and a few others were compared. In their results, the slow convergence of the multiplicative updating was confirmed, and the ALS method in Sect. 3.3 was shown to fail to converge in many cases. Among all the methods tested, the HALS/RRI method and the ANLS/BPP method showed the fastest overall convergence.

Further comparisons are presented in Gillis and Glineur [37] and Hsieh and Dhillon [48] where the authors proposed acceleration methods for the HALS/RRI method. Their comparisons show that the HALS/RRI method or the accelerated versions converge the fastest among all methods tested. Korattikara et al. [62] demonstrated an effective approach to accelerate the ANLS/BPP method. Overall, the HALS/RRI method, the ANLS/BPP method, and their accelerated versions show the state-of-the-art performance in the experimental comparisons.

Comparison results of algorithms for NCP are provided in [60]. Interestingly, the ANLS/BPP method showed faster convergence than the HALS/RRI method in the tensor factorization case. Further investigations and experimental evaluations of the NCP algorithms are needed to fully explain these observations.

7 Efficient NMF updating: algorithms

In practice, we often need to update a factorization with a slightly modified condition or some additional data. We consider two scenarios where an existing factorization needs to be efficiently updated to a new factorization. Importantly, the unified view of the NMF algorithms presented in earlier chapters provides useful insights when we choose algorithmic components for updating. Although we focus on the NMF updating here, similar updating schemes can be developed for NCP as well.

7.1 Updating NMF with an increased or decreased K

NMF algorithms discussed in Sects. 2 and 3 assume that K, the reduced dimension, is provided as an input. In practical applications, however, prior knowledge on K might not be available, and a proper value for K has to be determined from data. To determine K from data, typically NMFs are computed for several different K values and then the best K is chosen according to some criterion [10,33,49]. In this case, computing several NMFs each time from scratch would be very expensive, and therefore it is desired to develop an algorithm to efficiently update an already computed factorization. We propose an algorithm for this task in this subsection.

Suppose we have computed $\mathbf{W}_{old} \in \mathbb{R}_+^{M \times K_1}$ and $\mathbf{H}_{old} \in \mathbb{R}_+^{N \times K_1}$ as a solution of (2) with $K = K_1$. For $K = K_2$ which is close to K_1, we are to compute new factors $\mathbf{W}_{new} \in \mathbb{R}_+^{M \times K_2}$ and $\mathbf{H}_{new} \in \mathbb{R}_+^{N \times K_2}$ as a minimizer of (2). Let us first consider the $K_2 > K_1$ case, which is shown in Fig. 2. Each of $\mathbf{W}_{new}$ and $\mathbf{H}_{new}$ in this case contains $K_2 - K_1$ additional columns compared to $\mathbf{W}_{old}$ and $\mathbf{H}_{old}$. A natural strategy is to initialize new factor matrices by recycling $\mathbf{W}_{old}$ and $\mathbf{H}_{old}$ as

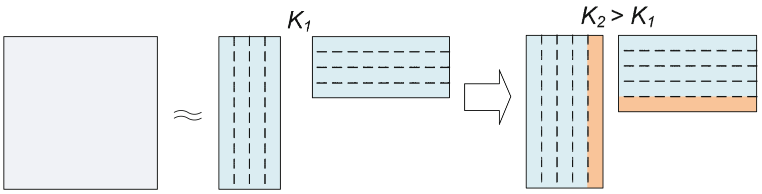

Fig. 2 Updating NMF with an increased K

$$\mathbf{W}_{new} = [\,\mathbf{W}_{old}\ \mathbf{W}_{add}\,] \text{ and } \mathbf{H}_{new} = [\,\mathbf{H}_{old}\ \mathbf{H}_{add}\,], \tag{64}$$

where $\mathbf{W}_{add} \in \mathbb{R}_{+}^{M\times(K_2-K_1)}$ and $\mathbf{H}_{add} \in \mathbb{R}_{+}^{N\times(K_2-K_1)}$ are generated with, e.g., random nonnegative entries. Using (64) as initial values, we can execute an NMF algorithm to find the solution of (2). Since $\mathbf{W}_{old}$ and $\mathbf{H}_{old}$ already approximate $\mathbf{A}$, this warm-start strategy is expected to be more efficient than computing everything from scratch.

We further improve this strategy based on the following observation. Instead of initializing $\mathbf{W}_{add}$ and $\mathbf{H}_{add}$ with random entries, we can compute $\mathbf{W}_{add}$ and $\mathbf{H}_{add}$ that approximately factorize the residual matrix, i.e., $\mathbf{A} - \mathbf{W}_{old}\mathbf{H}_{old}^T$. This can be done by solving the following problem:

$$(\mathbf{W}_{add}, \mathbf{H}_{add}) \leftarrow$$

$$\underset{\substack{\mathbf{W} \in \mathbb{R}^{M\times(K_2-K_1)} \\ \mathbf{H} \in \mathbb{R}^{N\times(K_2-K_1)}}}{\arg\min} \left\| (\mathbf{A} - \mathbf{W}_{old}\mathbf{H}_{old}^T) - \mathbf{W}\mathbf{H}^T \right\|_F^2 \tag{65}$$

$$\text{subject to } \mathbf{W} \geq 0, \mathbf{H} \geq 0.$$

Problem (65) need not be solved very accurately. Once an approximate solution is obtained, it is used to initialize $\mathbf{W}_{new}$ and $\mathbf{H}_{new}$ in (64) and then an NMF algorithm is executed for entire matrices $\mathbf{W}_{new}$ and $\mathbf{H}_{new}$.

When $K_2 < K_1$, we need less columns in $\mathbf{W}_{new}$ and $\mathbf{H}_{new}$ than in $\mathbf{W}_{old}$ and $\mathbf{H}_{old}$. For a good initialization of $\mathbf{W}_{new}$ and $\mathbf{H}_{new}$, we need to choose the columns from $\mathbf{W}_{old}$ and $\mathbf{H}_{old}$. Observing that

$$\mathbf{A} \approx \mathbf{W}_{old}\mathbf{H}_{old}^T = \sum_{k=1}^{K_1} (\mathbf{W}_{old})_{\cdot k} (\mathbf{H}_{old})_{\cdot k}^T, \tag{66}$$

K_2 columns can be selected as follows. Let δ_k be the squared Frobenius norm of the kth rank-one matrix in (66), given as

$$\delta_k = \| (\mathbf{W}_{old})_{\cdot k} (\mathbf{H}_{old})_{\cdot k}^T \|_F^2 = \| (\mathbf{W}_{old})_{\cdot k} \|_2^2 \| (\mathbf{H}_{old})_{\cdot k} \|_2^2.$$

We then take the largest K_2 values from $\delta_1, \ldots, \delta_{K_1}$ and use corresponding columns of $\mathbf{W}_{old}$ and $\mathbf{H}_{old}$ as initializations for $\mathbf{W}_{new}$ and $\mathbf{H}_{new}$.

Summarizing the two cases, an algorithm for updating NMF with an increased or decreased K value is presented in Algorithm 2. Note that the HALS/RRI method is chosen for Step 2: Since the new entries appear as column blocks (see Fig. 2), the HALS/RRI method is an optimal choice. For Step 2, although any algorithm may be chosen, we have adopted the HALS/RRI method for our experimental evaluation in Sect. 8.1.

Algorithm 2 Updating NMF with Increased or Decreased K Values

Input: $\mathbf{A} \in \mathbb{R}_+^{M\times N}$, $\left(\mathbf{W}_{old} \in \mathbb{R}_+^{M\times K_1}, \mathbf{H}_{old} \in \mathbb{R}_+^{N\times K_1}\right)$ as a minimizer of (2), and K_2.

Output: $\left(\mathbf{W}_{new} \in \mathbb{R}_+^{M\times K_2}, \mathbf{H}_{new} \in \mathbb{R}_+^{N\times K_2}\right)$ as a minimizer of (2).

```
1: if $K_2 > K_1$ then
2:     Approximately solve (65) with the HALS/RRI method to find $(\mathbf{W}_{add}, \mathbf{H}_{add})$.
3:     Let $\mathbf{W}_{new} \leftarrow [\, \mathbf{W}_{old}\ \ \mathbf{W}_{add}\,]$ and $\mathbf{H}_{new} \leftarrow [\, \mathbf{H}_{old}\ \ \mathbf{H}_{add}\,]$.
4: end if
5: if $K_2 < K_1$ then
6:     For $k = 1, \ldots, K_1$, let
           $\delta_k = \| (\mathbf{W}_{old})_{\cdot k} \|_2^2 \| (\mathbf{H}_{old})_{\cdot k} \|_2^2.$
7:     Let $J$ be the indices corresponding to the $K_2$ largest values of $\delta_1, \ldots, \delta_{K_1}$.
8:     Let $\mathbf{W}_{new}$ and $\mathbf{H}_{new}$ be the submatrices of $\mathbf{W}_{old}$ and $\mathbf{H}_{old}$ obtained from the columns indexed by $J$.
9: end if
10: Using $\mathbf{W}_{new}$ and $\mathbf{H}_{new}$ as initial values, execute an NMF algorithm to compute NMF of $\mathbf{A}$.
```

7.2 Updating NMF with incremental data

In applications such as video analysis and mining of text stream, we have to deal with dynamic data where new data keep coming in and obsolete data get discarded. Instead of completely recomputing the factorization after only a small portion of data are updated, an efficient algorithm needs to be designed to update NMF. Let us first consider a case that new data are observed, as shown in Fig. 3. Suppose we have computed $\mathbf{W}_{old} \in \mathbb{R}_+^{M\times K}$ and $\mathbf{H}_{old} \in \mathbb{R}_+^{N\times K}$ as a minimizer of (2) for $\mathbf{A} \in \mathbb{R}_+^{M\times N}$. New data, $\Delta\mathbf{A} \in \mathbb{R}_+^{M\times \Delta N}$, are placed in the last columns of a new matrix as $\tilde{\mathbf{A}} = [\mathbf{A}\Delta\mathbf{A}]$. Our goal is to efficiently compute the updated NMF

$$\tilde{\mathbf{A}} = [\mathbf{A}\Delta\mathbf{A}] \approx \mathbf{W}_{new}\mathbf{H}_{new}^T,$$

where $\mathbf{W}_{new} \in \mathbb{R}_+^{M\times K}$ and $\mathbf{H}_{new} \in \mathbb{R}_+^{(N+\Delta N)\times K}$.

The following strategy we propose is simple but efficient. Since columns in $\mathbf{W}_{old}$ form a basis whose nonnegative combinations approximate $\mathbf{A}$, it is reasonable to use $\mathbf{W}_{old}$ to initialize $\mathbf{W}_{new}$. Similarly, $\mathbf{H}_{new}$ is initialized as $\begin{bmatrix} \mathbf{H}_{old} \\ \Delta\mathbf{H} \end{bmatrix}$ where the first part, $\mathbf{H}_{old}$, is obtained from the existing factorization. A new coefficient submatrix, $\Delta\mathbf{H} \in \mathbb{R}_+^{\Delta N\times K}$, is needed to represent the coefficients for new data. Although it is possible to initialize $\Delta\mathbf{H}$ with random entries, an improved approach is to solve the following NLS problem:

$$\Delta\mathbf{H} \leftarrow \underset{\mathbf{H}\in\mathbb{R}^{\Delta N\times K}}{\arg\min} \ \|\mathbf{W}_{old}\mathbf{H}^T - \Delta\mathbf{A}\|_F^2 \text{ s.t. } \mathbf{H} \geq 0. \tag{67}$$

Using these initializations, we can then execute an NMF algorithm to find an optimal solution for $\mathbf{W}_{new}$ and $\mathbf{H}_{new}$. Various algorithms for the NLS problem, discussed in Sect. 4, maybe used to solve (67). In order to achieve optimal efficiency, due to the fact that the number of rows of $\Delta\mathbf{H}^T$ is usually small, the block principal pivoting algorithm is one of the most efficient method as demonstrated in [59]. We summarize this method for updating NMF with incremental data in Algorithm 3.

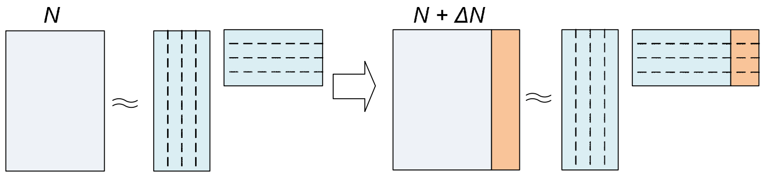

Fig. 3 Updating NMF with incremental data

Algorithm 3 Updating NMF with Incremental Data

```
Input: $\mathbf{A} \in \mathbb{R}_+^{M\times N}$, $\left(\mathbf{W}_{old} \in \mathbb{R}_+^{M\times K}, \mathbf{H}_{old} \in \mathbb{R}_+^{N\times K}\right)$ as a solution of Eq. (2), and $\Delta\mathbf{A} \in \mathbb{R}_+^{M\times \Delta N}$.
Output: $\mathbf{W}_{new} \in \mathbb{R}_+^{M\times K}$ and $\mathbf{H}_{new} \in \mathbb{R}_+^{(N+\Delta N)\times K}$ as a solution of Eq. (2).
1: Solve the following NLS problem:
       $\Delta\mathbf{H} \leftarrow \underset{\mathbf{H}\in\mathbb{R}^{\Delta N\times K}}{\arg\min} \|\mathbf{W}_{old}\mathbf{H}^T - \Delta\mathbf{A}\|_F^2$ s.t. $\mathbf{H} \geq 0$.
2: Let $\mathbf{W}_{new} \leftarrow \mathbf{W}_{old}$ and $\mathbf{H}_{new} \leftarrow \begin{bmatrix}\mathbf{H}_{old} \\ \Delta\mathbf{H}\end{bmatrix}$.
3: Using $\mathbf{W}_{new}$ and $\mathbf{H}_{new}$ as initial values, execute an NMF algorithm to compute NMF of $[\,\mathbf{A}\ \ \Delta\mathbf{A}\,]$.
```

Deleting obsolete data is easier. If $\mathbf{A} = [\Delta\mathbf{A}\tilde{\mathbf{A}}]$ where $\Delta\mathbf{A} \in \mathbb{R}_+^{M\times \Delta N}$ is to be discarded, we similarly divide $\mathbf{H}_{old}$ as $\mathbf{H}_{old} = \begin{bmatrix}\Delta\mathbf{H} \\ \tilde{\mathbf{H}}_{old}\end{bmatrix}$. We then use $\mathbf{W}_{old}$ and $\tilde{\mathbf{H}}_{old}$ to initialize $\mathbf{W}_{new}$ and $\mathbf{H}_{new}$ and execute an NMF algorithm to find a minimizer of (2).

8 Efficient NMF updating: experiments and applications

We provide the experimental validations of the effectiveness of Algorithms 2 and 3 and show their applications. The computational efficiency was compared on dense and sparse synthetic matrices as well as on real-world data sets. All the experiments were executed with MATLAB on a Linux machine with 2GHz Intel Quad-core processor and 4GB memory.

8.1 Comparisons of NMF updating methods for varying K

We compared Algorithm 2 with two alternative methods for updating NMF. The first method is to compute NMF with $K = K_2$ from scratch using the HALS/RRI algorithm, which we denote as ‘recompute’ in our figures. The second method, denoted as ‘warm-restart’ in the figures, computes the new factorization as follows. If $K_2 > K_1$, it generates $\mathbf{W}_{add} \in \mathbb{R}_+^{M\times(K_2-K_1)}$ and $\mathbf{H}_{add} \in \mathbb{R}_+^{N\times(K_2-K_1)}$ using random entries to initialize $\mathbf{W}_{new}$ and $\mathbf{H}_{new}$ as in (64). If $K_2 < K_1$, it randomly selects K_2 pairs of columns from $\mathbf{W}_{old}$ and $\mathbf{H}_{old}$ to initialize the new factors. Using these initializations, ‘warm-restart’ executes the HALS/RRI algorithm to finish the NMF computation.

Synthetic data sets and performance comparisons on them are as follows. We created both dense and sparse matrices. For dense matrices, we generated $\mathbf{W} \in \mathbb{R}_+^{M\times K}$ and $\mathbf{H} \in \mathbb{R}_+^{N\times K}$ with random entries and computed $\mathbf{A} = \mathbf{W}\mathbf{H}^T$. Then, Gaussian noise was added to the elements of $\mathbf{A}$ where the noise has zero mean and standard deviation is 5 % of the average magnitude of elements in $\mathbf{A}$. All negative elements after adding the noise were set to zero.

We generated a 600×600 dense matrix with $K = 80$. For sparse matrices, we first generated $\mathbf{W} \in \mathbb{R}_+^{M \times K}$ and $\mathbf{H} \in \mathbb{R}_+^{N \times K}$ with 90 % sparsity and computed $\mathbf{A} = \mathbf{W}\mathbf{H}^T$. We then used a soft-thresholding operation to obtain a sparse matrix $\mathbf{A}$,[1] and the resulting matrix had 88.2 % sparsity. We generated a synthetic sparse matrix of size $3,000 \times 3,000$.[2] In order to observe efficiency in updating, an NMF with $K_1 = 60$ was first computed and stored. We then computed NMFs with $K_2 = 50, 65$, and 80. The plots of relative error vs. execution time for all three methods are shown in Fig. 4. Our proposed method achieved faster convergence compared to 'warm-restart' and 'recompute', which sometimes required several times more computation to achieve the same accuracy as our method. The advantage of the proposed method can be seen in both dense and sparse cases.

We have also used four real-world data sets for our comparisons. From the Topic Detection and Tracking 2 (TDT2) text corpus,[3] we selected 40 topics to create a sparse term-document matrix of size $19,009 \times 3,087$. From the 20 Newsgroups data set,[4] a sparse term-document matrix of size $7,845 \times 11,269$ was obtained after removing keywords and documents with frequency less than 20. The AT&T facial image database[5] produced a dense matrix of size $10,304 \times 400$. The images in the CMU PIE database[6] were resized to 64×64 pixels, and we formed a dense matrix of size $4,096 \times 11,554$.[7] We focused on the case when K increases, and the results are reported in Fig. 5. As with the synthetic data sets, our proposed method was shown to be the most efficient among the methods we tested.

8.2 Applications of NMF updating for varying K

Algorithm 2 can be used to determine the reduced dimension, K, from data. Our first example, shown in Fig. 6a, is determining a proper K value that represents the number of clusters. Using NMF as a clustering method [57,63], Brunet et al. [10] proposed to select the number of clusters by computing NMFs with multiple initializations for various K values and then evaluating the dispersion coefficients (See [10,52,57] for more details). We took the MNIST digit image database [65] and used 500 images with 28×28 pixels from each of the digits 6, 7, 8, and 9. The resulting data matrix was of size $784 \times 2,000$. We computed NMFs for $K = 3, 4, 5$, and 6 with 50 different initializations for each K. The top of Fig. 6a shows that $K = 4$ can be correctly determined from the point where the dispersion coefficient starts to drop. The bottom of Fig. 6a shows the box-plot of the total execution time needed by Algorithm 2, 'recompute', and 'warm-restart'. We applied the same stopping criterion in (62) for all three methods.

Further applications of Algorithm 2 are shown in Fig. 6b, c. Figure 6b demonstrates a process of probing the approximation errors of NMF with various K values. With $K = 20, 40, 60$ and 80, we generated 600×600 synthetic dense matrices as described in Sect. 8.1. Then, we computed NMFs with Algorithm 2 for K values ranging from 10 to 160 with a step size 5. The relative objective function values with respect to K are shown in Fig. 6b. In each of the cases where $K = 20, 40, 60$, and 80, we were able to determine the correct K value by choosing a point where the relative error stopped decreasing significantly.

[1] For each element a_{ij}, we used $a_{ij} \leftarrow \max(a_{ij} - 2, 0)$.

[2] We created a larger matrix for the sparse case to clearly illustrate the relative efficiency.

[3] http://projects.ldc.upenn.edu/TDT2/.

[4] http://people.csail.mit.edu/jrennie/20Newsgroups/.

[5] http://www.cl.cam.ac.uk/research/dtg/attarchive/facedatabase.html.

[6] http://www.ri.cmu.edu/projects/project_418.html.

[7] http://www.zjucadcg.cn/dengcai/Data/FaceData.html.

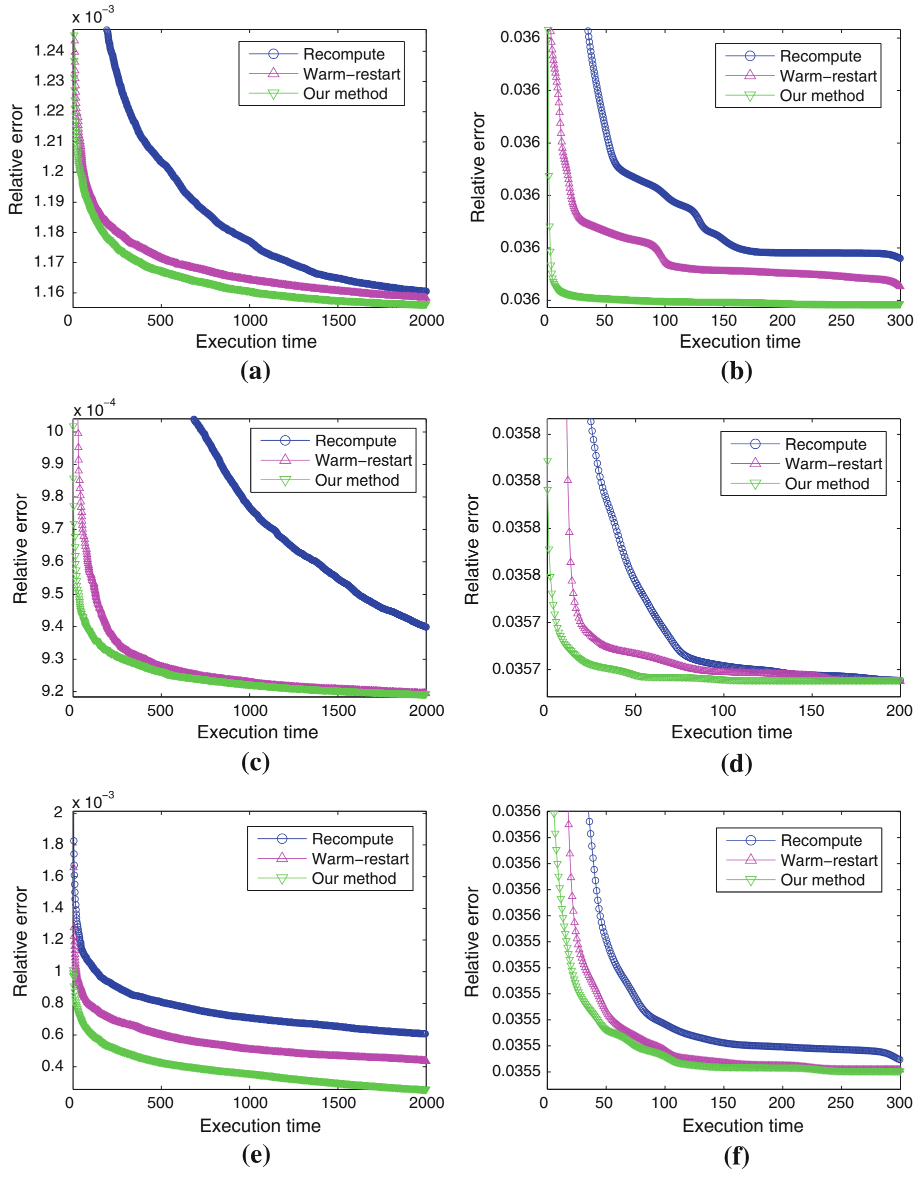

Fig. 4 Comparisons of updating and recomputing methods for NMF when K changes, using synthetic matrices. The relative error represents $\|\mathbf{A} - \mathbf{W}\mathbf{H}^T\|_F / \|\mathbf{A}\|_F$, and time was measured in seconds. The dense matrix was of size 600×600, and the sparse matrix was of size $3,000 \times 3,000$. See text for more details. **a** Dense, $K : 60 \to 50$. **b** Sparse, $K : 60 \to 50$. **c** Dense, $K : 60 \to 65$. **d** Sparse, $K : 60 \to 65$. **e** Dense, $K : 60 \to 80$. **f** Sparse, $K : 60 \to 80$

Figure 6c demonstrates the process of choosing K for a classification purpose. Using the $10,304 \times 400$ matrix from the AT&T facial image database, we computed NMF to generate a K dimensional representation of each image, taken from each row of $\mathbf{H}$. We then trained a nearest neighbor classifier using the reduced-dimensional representations [83]. To determine

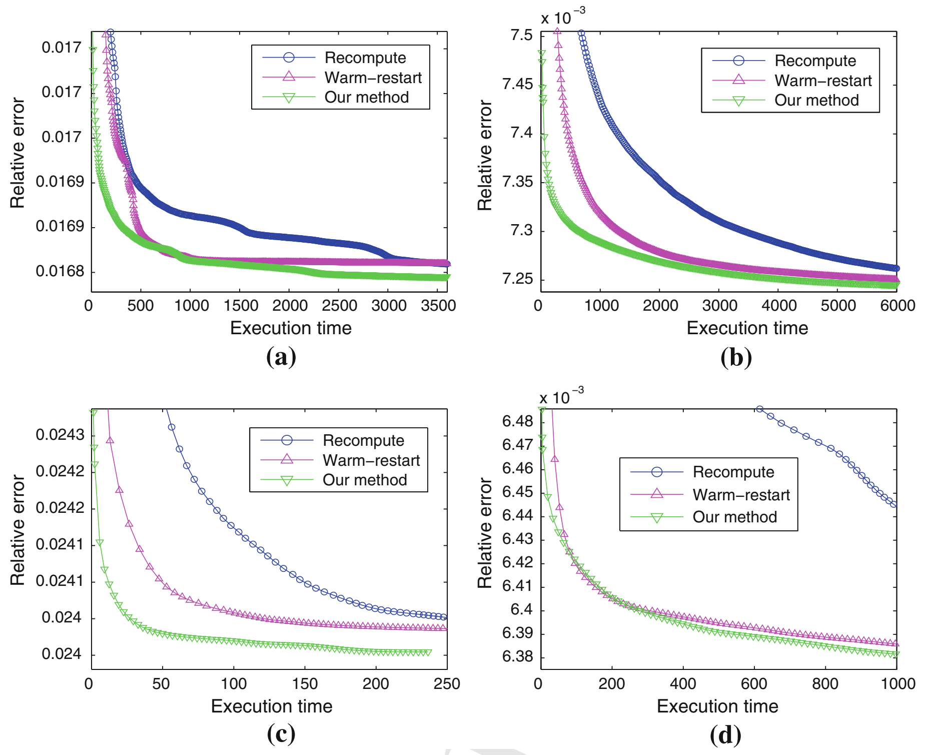

Fig. 5 Comparisons of updating and recomputing methods for NMF when K changes, using real-world data sets. The relative error represents $\|\mathbf{A} - \mathbf{W}\mathbf{H}^T\|_F/\|\mathbf{A}\|_F$, and time was measured in seconds. The AT&T and the PIE data sets were dense matrices of size $10,304 \times 400$ and $4,096 \times 11,554$, respectively. The TDT2 and the 20 Newsgroup data sets were sparse matrices of size $19,009 \times 3,087$ and $7,845 \times 11,269$, respectively. **a** AT&T, dense, $K : 80 \rightarrow 100$, **b** PIE, dense, $K : 80 \rightarrow 100$, **c** TDT2, sparse, $K : 160 \rightarrow 200$, **d** 20 Newsgroup, sparse, $K : 160 \rightarrow 200$

the best K value, we performed the 5-fold cross validation: Each time a data matrix of size $10,304 \times 320$ was used to compute $\mathbf{W}$ and $\mathbf{H}$, and the reduced-dimensional representations for the test data $\tilde{\mathbf{A}}$ were obtained by solving a NLS problem, $\min_{\tilde{\mathbf{H}} \geq 0} \left\| \tilde{\mathbf{A}} - \mathbf{W}\tilde{\mathbf{H}} \right\|_F^2$. Classification errors on both training and testing sets are shown in Fig. 6c. Five paths of training and testing errors are plotted using thin lines, and the averaged training and testing errors are plotted using thick lines. Based on the figure, we chose $K = 13$ since the testing error barely decreased beyond the point whereas the training error approached zero.

8.3 Comparisons of NMF updating methods for incremental data

We also tested the effectiveness of Algorithm 3. We created a $1,000 \times 500$ dense matrix $\mathbf{A}$ as described in Sect. 8.1 with $K = 100$. An initial NMF with $K = 80$ was computed and stored. Then, an additional data set of size $1,000 \times 10$, $1,000 \times 20$, or $1,000 \times 50$ was appended, and we computed the updated NMF with several methods as follows. In addition to Algorithm 3, we considered four alternative methods. A naive approach that computes the entire NMF from scratch is denoted as 'recompute'. An approach that initializes a new

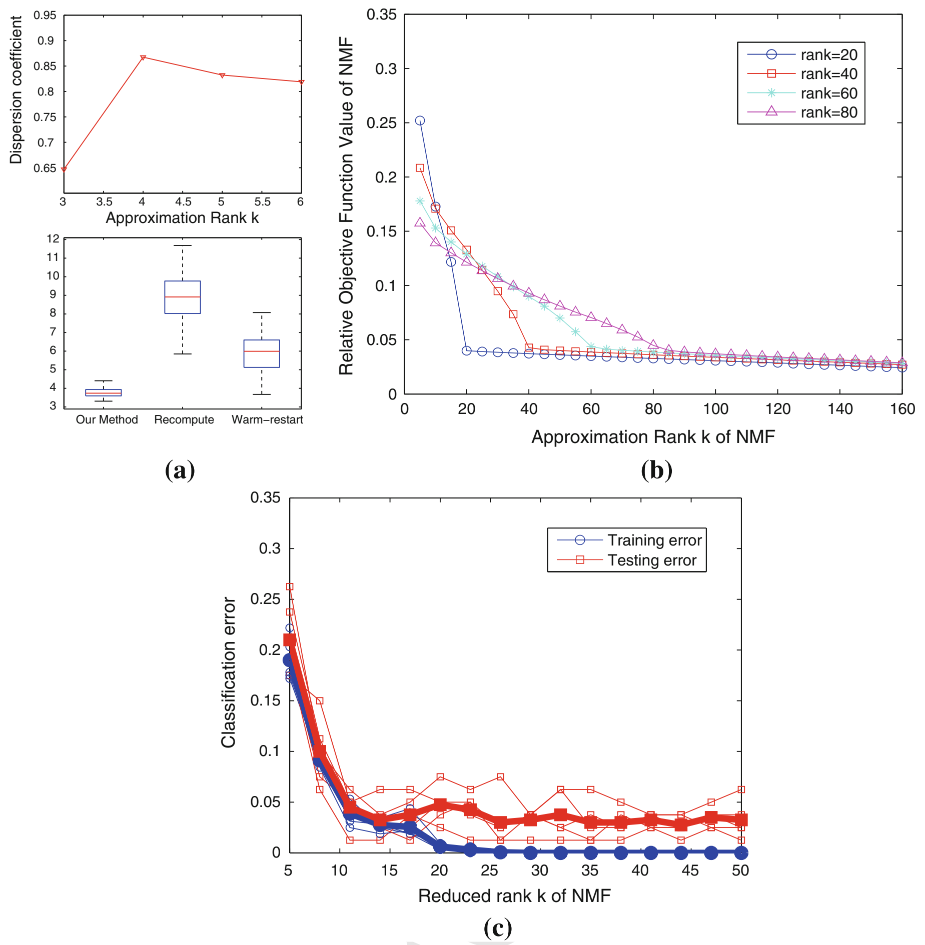

Fig. 6 **a** *Top* Dispersion coefficients (see [10,57]) obtained by using 50 different initializations for each K, *Bottom* Execution time needed by our method, 'recompute', and 'warm-restart'. **b** Relative errors for various K values on data sets created with $K = 20, 40, 60$ and 80. **c** Classification errors on training and testing data sets of AT&T facial image database using 5-fold cross validation

coefficient matrix as $\mathbf{H}_{new} = \begin{bmatrix} \mathbf{H}_{old} \\ \Delta\mathbf{H} \end{bmatrix}$ where $\Delta\mathbf{H}$ is generated with random entries is denoted as 'warm-restart'. The incremental NMF algorithm (INMF) [11] as well as the online NMF algorithm (ONMF) [13] were also included in the comparisons. Figure 7 shows the execution results, where our proposed method outperforms other methods tested.

9 Conclusion and discussion

We have reviewed algorithmic strategies for computing NMF and NTF from a unifying perspective based on the BCD framework. The BCD framework for NMF and NTF enables simplified understanding of several successful algorithms such as the alternating nonnegative least squares (ANLS) and the hierarchical alternating least squares (HALS) methods. Based

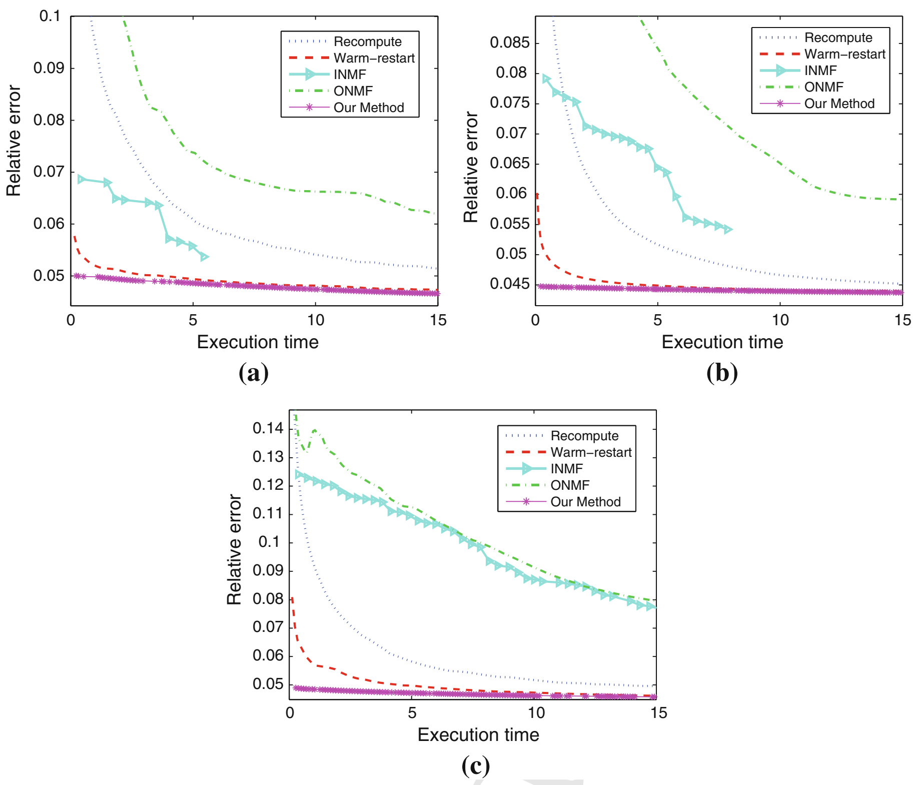

Fig. 7 Comparisons of NMF updating methods for incremental data. Given a $1,000 \times 500$ matrix and a corresponding NMF, ΔN additional data items were appended and the NMF was updated. The relative error represents $\|\mathbf{A} - \mathbf{W}\mathbf{H}^T\|_F / \|\mathbf{A}\|_F$, and time was measured in seconds. **a** $\Delta N = 10$, **b** $\Delta N = 20$, **c** $\Delta N = 50$

on the BCD framework, the theoretical convergence properties of the ANLS and the HALS methods are readily explained. We have also summarized how previous algorithms that do not fit in the BCD framework differ from the BCD-based methods. With insights from the unified view, we proposed efficient algorithms for updating NMF both for the cases that the reduced dimension varies and that data are incrementally added or discarded.

There are many other interesting aspects of NMF that are not covered in this paper. Depending on the probabilistic model of the underlying data, NMF can be formulated with various divergences. Formulations and algorithms based on Kullback-Leibler divergence [67,79], Bregman divergence [24,68], Itakura-Saito divergence [29], and Alpha and Beta divergences [21,22] have been developed. For discussion on nonnegative rank as well as the geometric interpretation of NMF, see Lin and Chu [72], Gillis [34], and Donoho and Stodden [27]. NMF has been also studied from the Bayesian statistics point of view: See Schmidt et al. [78] and Zhong and Girolami [88]. In the data mining community, variants of NMF such as convex and semi-NMFs [25,75], orthogonal tri-NMF [26], and group-sparse NMF [56] have been proposed, and using NMF for clustering has been shown to be successful [12,57,63]. For an overview on the use of NMF in bioinformatics, see Devarajan [23] and references therein. Cichocki et al.'s book [22] explains the use of NMF for signal processing. See Chu and Plemmons [17], Berry et al. [4], and Cichocki et al. [22] for earlier surveys on NMF. See also Ph.D. dissertations on NMF algorithms and applications [34,44,55].

References

1. Acar, E., Yener, B.: Unsupervised multiway data analysis: a literature survey. IEEE Trans. Knowl. Data Eng. **21**(1), 6–20 (2009)
2. Bellavia, S., Macconi, M., Morini, B.: An interior point newton-like method for non-negative least-squares problems with degenerate solution. Numer. Linear Algebra Appl. **13**(10), 825–846 (2006)
3. Berman, A., Plemmons, R.J.: Nonnegative matrices in the mathematical sciences. Society for Industrial and Applied Mathematics, Philadelphia (1994)
4. Berry, M., Browne, M., Langville, A., Pauca, V., Plemmons, R.: Algorithms and applications for approximate nonnegative matrix factorization. Comput. Stat. Data Anal. **52**(1), 155–173 (2007)
5. Bertsekas, D.P.: Nonlinear programming. Athena Scientific (1999)
6. Biggs, M., Ghodsi, A., Vavasis, S.: Nonnegative matrix factorization via rank-one downdate. In: Proceedings of the 25th International Conference on, Machine Learning, pp. 64–71 (2008)
7. Birgin, E., Martínez, J., Raydan, M.: Nonmonotone spectral projected gradient methods on convex sets. SIAM J. Optim. **10**(4), 1196–1211 (2000)
8. Björck, Å.: Numerical Methods for Least Squares Problems. Society for Industrial and Applied Mathematics, Philadelphia (1996)
9. Bro, R., De Jong, S.: A fast non-negativity-constrained least squares algorithm. J. Chemom. **11**, 393–401 (1997)
10. Brunet, J., Tamayo, P., Golub, T., Mesirov, J.: Metagenes and molecular pattern discovery using matrix factorization. Proc. Natal. Acad. Sci. **101**(12), 4164–4169 (2004)
11. Bucak, S., Gunsel, B.: Video content representation by incremental non-negative matrix factorization. In: Proceedings of the 2007 IEEE International Conference on Image Processing (ICIP), vol. 2, pp. II-113–II-116 (2007)
12. Cai, D., He, X., Han, J., Huang, T.: Graph regularized nonnegative matrix factorization for data representation. IEEE Trans. Pattern Anal. Mach. Intell. **33**(8), 1548–1560 (2011)
13. Cao, B., Shen, D., Sun, J.T., Wang, X., Yang, Q., Chen, Z.: Detect and track latent factors with online nonnegative matrix factorization. In: Proceedings of the 20th International Joint Conference on Artifical, Intelligence, pp. 2689–2694 (2007)
14. Carroll, J.D., Chang, J.J.: Analysis of individual differences in multidimensional scaling via an n-way generalization of "eckart-young" decomposition. Psychometrika **35**(3), 283–319 (1970)
15. Chen, D., Plemmons, R.J.: Nonnegativity constraints in numerical analysis. In: Proceedings of the Symposium on the Birth of Numerical Analysis, Leuven Belgium, pp. 109–140 (2009)
16. Chen, S.S., Donoho, D.L., Saunders, M.A.: Atomic decomposition by basis pursuit. SIAM Rev. **43**(1), 129–159 (2001)
17. Chu, M., Plemmons, R.: Nonnegative matrix factorization and applications. IMAGE: Bull. Int. Linear Algebra Soc. **34**, 2–7 (2005)
18. Chu, M.T., Lin, M.M.: Low-dimensional polytope approximation and its applications to nonnegative matrix factorization. SIAM J. Sci. Comput. **30**(3), 1131–1155 (2008)
19. Cichocki, A., Phan, A.H.: Fast local algorithms for large scale nonnegative matrix and tensor factorizations. IEICE Trans. Fundam. Electron. Commun. Comput. Sci. **E92-A**(3), 708–721 (2009)
20. Cichocki, A., Zdunek, R., Amari, S.I.: Hierarchical ALS algorithms for nonnegative matrix and 3d tensor factorization. In: Lecture Notes in Computer Science, vol. 4666, pp. 169–176. Springer (2007)
21. Cichocki, A., Zdunek, R., Choi, S., Plemmons, R., Amari, S.-I.: Nonnegative tensor factorization using alpha and beta divergencies. In: Proceedings of the 32nd International Conference on Acoustics, Speech, and Signal Processing (ICASSP), Honolulu, April 2007, vol. 3, pp. III-1393–III-1396 (2007)
22. Cichocki, A., Zdunek, R., Phan, A.H., Amari, S.I.: Nonnegative Matrix and Tensor Factorizations: Applications to Exploratory Multi-Way Data Analysis and Blind Source Separation. Wiley, West Sussex (2009)
23. Devarajan, K.: Nonnegative matrix factorization: an analytical and interpretive tool in computational biology. PLoS Comput. Biol. **4**(7), e1000,029 (2008)

24. Dhillon, I., Sra, S.: Generalized nonnegative matrix approximations with bregman divergences. In: Advances in Neural Information Processing Systems 18, pp. 283–290. MIT Press (2006)
25. Ding, C., Li, T., Jordan, M.: Convex and semi-nonnegative matrix factorizations. IEEE Trans. Pattern Anal. Mach. Intell. **32**(1), 45–559 (2010)
26. Ding, C., Li, T., Peng, W., Park, H.: Orthogonal nonnegative matrix tri-factorizations for clustering. In: Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 126–135 (2006)
27. Donoho, D., Stodden, V.: When does non-negative matrix factorization give a correct decomposition into parts? In: Advances in Neural Information Processing Systems 16. MIT Press (2004)
28. Drake, B., Kim, J., Mallick, M., Park, H.: Supervised Raman spectra estimation based on nonnegative rank deficient least squares. In: Proceedings of the 13th International Conference on Information Fusion, Edinburgh, UK (2010)
29. Févotte, C., Bertin, N., Durrieu, J.: Nonnegative matrix factorization with the Itakura-Saito divergence: With application to music analysis. Neural Comput. **21**(3), 793–830 (2009)
30. Figueiredo, M.A.T., Nowak, R.D., Wright, S.J.: Gradient projection for sparse reconstruction: application to compressed sensing and other inverse problems. IEEE J. Sel. Top. Signal Process. **1**(4), 586–597 (2007)
31. Franc, V., Hlavac, V., Navara, M.: Sequential coordinate-wise algorithm for the non-negative least squares problem. In: Proceedings of the 11th International Conference on Computer Analysis of Images and Patterns, pp. 407–414 (2005)
32. Friedlander, M.P., Hatz, K.: Computing nonnegative tensor factorizations. Comput. Optim. Appl. **23**(4), 631–647 (2008)
33. Frigyesi, A., Höglund, M.: Non-negative matrix factorization for the analysis of complex gene expression data: identification of clinically relevant tumor subtypes. Cancer Inform. **6**, 275–292 (2008)
34. Gillis, N.: Nonnegative matrix factorization complexity, algorithms and applications. Ph.D. thesis, Université catholique de Louvain (2011)
35. Gillis, N., Glineur, F.: Nonnegative factorization and the maximum edge biclique problem. CORE Discussion Paper 2008/64, Universite catholique de Louvain (2008)
36. Gillis, N., Glineur, F.: Using underapproximations for sparse nonnegative matrix factorization. Pattern Recognit. **43**(4), 1676–1687 (2010)
37. Gillis, N., Glineur, F.: Accelerated multiplicative updates and hierarchical als algorithms for nonnegative matrix factorization. Neural Comput. **24**(4), 1085–1105 (2012)
38. Gillis, N., Glineur, F.: A multilevel approach for nonnegative matrix factorization. J. Comput. Appl. Math. **236**, 1708–1723 (2012)
39. Golub, G., Van Loan, C.: Matrix Computations. Johns Hopkins University Press, Baltimore (1996)
40. Gonzalez, E.F., Zhang, Y.: Accelerating the lee-seung algorithm for non-negative matrix factorization. Department of Computational and Applied Mathematics, Rice University, Technical report (2005)
41. Grippo, L., Sciandrone, M.: On the convergence of the block nonlinear gauss-seidel method under convex constraints. Oper. Res. Lett. **26**(3), 127–136 (2000)
42. Han, L., Neumann, M., Prasad, U.: Alternating projected Barzilai-Borwein methods for nonnegative matrix factorization. Electron. Trans. Numer. Anal. **36**, 54–82 (2009)
43. Harshman, R.A.: Foundations of the parafac procedure: models and conditions for an "explanatory" multi-modal factor analysis. In: UCLA Working Papers in Phonetics, vol. 16, pp. 1–84 (1970)
44. Ho, N.D.: Nonnegative matrix factorization algorithms and applications. Ph.D. thesis, Univ. Catholique de Louvain (2008)
45. Horn, R.A., Johnson, C.R.: Matrix Analysis. Cambridge University Press, Cambridge (1990)
46. Horst, R., Pardalos, P., Van Thoai, N.: Introduction to Global Optimization. Kluwer, Berlin (2000)
47. Hoyer, P.O.: Non-negative matrix factorization with sparseness constraints. J. Mach. Learn. Res. **5**, 1457–1469 (2004)
48. Hsieh, C.J., Dhillon, I.S.: Fast coordinate descent methods with variable selection for non-negative matrix factorization. In: Proceedings of the 17th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 1064–1072 (2011)
49. Hutchins, L., Murphy, S., Singh, P., Graber, J.: Position-dependent motif characterization using non-negative matrix factorization. Bioinformatics **24**(23), 2684–2690 (2008)
50. Júdice, J.J., Pires, F.M.: A block principal pivoting algorithm for large-scale strictly monotone linear complementarity problems. Comput. Oper. Res. **21**(5), 587–596 (1994)
51. Kim, D., Sra, S., Dhillon, I.S.: Fast Newton-type methods for the least squares nonnegative matrix approximation problem. In: Proceedings of the 2007 SIAM International Conference on Data Mining, pp. 343–354 (2007)
52. Kim, H., Park, H.: Sparse non-negative matrix factorizations via alternating non-negativity-constrained least squares for microarray data analysis. Bioinformatics **23**(12), 1495–1502 (2007)

53. Kim, H., Park, H.: Nonnegative matrix factorization based on alternating nonnegativity constrained least squares and active set method. SIAM J. Matrix Anal. Appl. **30**(2), 713–730 (2008)
54. Kim, H., Park, H., Eldén, L.: Non-negative tensor factorization based on alternating large-scale non-negativity-constrained least squares. In: Proceedings of IEEE 7th International Conference on Bioinformatics and, Bioengineering (BIBE07), vol. 2, pp. 1147–1151 (2007)
55. Kim, J.: Nonnegative Matrix and Tensor Factorizations, Least Squares Problems, and Applications. Ph.D. Thesis, Georgia Institute of Technology (2011)
56. Kim, J., Monteiro, R.D., Park, H.: Group Sparsity in Nonnegative Matrix Factorization. In: Proceedings of the 2012 SIAM International Conference on Data Mining, pp 851–862 (2012)
57. Kim, J., Park, H.: Sparse nonnegative matrix factorization for clustering. Technical report, Georgia Institute of Technology GT-CSE-08-01 (2008)
58. Kim, J., Park, H.: Toward faster nonnegative matrix factorization: a new algorithm and comparisons. In: Proceedings of the 8th IEEE International Conference on Data Mining (ICDM), pp. 353–362 (2008)
59. Kim, J., Park, H.: Fast nonnegative matrix factorization: an active-set-like method and comparisons. SIAM J. Sci. Comput. **33**(6), 3261–3281 (2011)
60. Kim, J., Park, H.: Fast nonnegative tensor factorization with an active-set-like method. In: High-Performance Scientific Computing: Algorithms and Applications, pp. 311–326. Springer (2012)
61. Kolda, T.G., Bader, B.W.: Tensor decompositions and applications. SIAM Rev. **51**(3), 455–500 (2009)
62. Korattikara, A., Boyles, L., Welling, M., Kim, J., Park, H.: Statistical optimization of non-negative matrix factorization. In: Proceedings of the 14th International Conference on Artificial Intelligence and Statistics (AISTATS), JMLR: W&CP, vol. 15, pp. 128–136 (2011)
63. Kuang, D., Ding, C., Park, H.: Symmetric nonnegative matrix factorization for graph clustering. In: Proceedings of 2012 SIAM International Conference on Data Mining, pp. 106–117 (2012)
64. Lawson, C.L., Hanson, R.J.: Solving Least Squares Problems. Prentice Hall, New Jersey (1974)
65. LeCun, Y., Bottou, L., Bengio, Y., Haffner, P.: Gradient-based learning applied to document recognition. Proc. IEEE **86**(11), 2278–2324 (1998)
66. Lee, D.D., Seung, H.S.: Learning the parts of objects by non-negative matrix factorization. Nature **401**(6755), 788–791 (1999)
67. Lee, D.D., Seung, H.S.: Algorithms for non-negative matrix factorization. In: Advances in Neural Information Processing Systems 13, pp. 556–562. MIT Press (2001)
68. Li, L., Lebanon, G., Park, H.: Fast bregman divergence nmf using taylor expansion and coordinate descent. In: Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining, pp. 307–315 (2012)
69. Li, S.Z., Hou, X., Zhang, H., Cheng, Q.: Learning spatially localized, parts-based representation. In: Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and, Pattern Recognition, vol. 1, pp. I-207–I-212 (2001)
70. Lin, C.: On the convergence of multiplicative update algorithms for nonnegative matrix factorization. IEEE Trans. Neural Netw. **18**(6), 1589–1596 (2007)
71. Lin, C.J.: Projected gradient methods for nonnegative matrix factorization. Neural Comput. **19**(10), 2756–2779 (2007)
72. Lin, M.M., Chu, M.T.: On the nonnegative rank of euclidean distance matrices. Linear Algebra Appl. **433**(3), 681–689 (2010)
73. Merritt, M., Zhang, Y.: Interior-point gradient method for large-scale totally nonnegative least squares problems. J. optim. Theory Appl. **126**(1), 191–202 (2005)
74. Paatero, P., Tapper, U.: Positive matrix factorization: a non-negative factor model with optimal utilization of error estimates of data values. Environmetrics **5**(1), 111–126 (1994)
75. Park, H., Kim, H.: One-sided non-negative matrix factorization and non-negative centroid dimension reduction for text classification. In: Proceedings of the 2006 Text Mining Workshop in the Tenth SIAM International Conference on Data Mining (2006)
76. Pauca, V.P., Piper, J., Plemmons, R.J.: Nonnegative matrix factorization for spectral data analysis. Linear Algebra Appl. **416**(1), 29–47 (2006)
77. Pauca, V.P., Shahnaz, F., Berry, M.W., Plemmons, R.J.: Text mining using non-negative matrix factorizations. In: Proceedings of the 2004 SIAM International Conference on Data Mining, pp. 452–456 (2004)
78. Schmidt, M.N., Winther, O., Hansen, L.K.: Bayesian non-negative matrix factorization. In: Proceedings of the 2009 International Conference on Independent Component Analysis and Signal Separation, Lecture Notes in Computer Science (LNCS), vol. 5441, pp. 540–547. Springer (2009)
79. Sra, S.: Block-iterative algorithms for non-negative matrix approximation. In: Proceedings of the 8th IEEE International Conference on Data Mining, pp. 1037–1042 (2008)
80. Tibshirani, R.: Regression shrinkage and selection via the lasso. J. R. Stat. Soc. Ser. B (Methodological) **58**(1), 267–288 (1996)

81. Van Benthem, M.H., Keenan, M.R.: Fast algorithm for the solution of large-scale non-negativity-constrained least squares problems. J. Chemom. **18**, 441–450 (2004)
82. Vavasis, S.A.: On the complexity of nonnegative matrix factorization. SIAM J. Optim. **20**(3), 1364–1377 (2009)
83. Weinberger, K., Saul, L.: Distance metric learning for large margin nearest neighbor classification. J. Mach. Learn. Res. **10**, 207–244 (2009)
84. Welling, M., Weber, M.: Positive tensor factorization. Pattern Recogn. Lett. **22**(12), 1255–1261 (2001)
85. Xu, W., Liu, X., Gong, Y.: Document clustering based on non-negative matrix factorization. In: Proceedings of the 26th Annual International ACM SIGIR Conference on Research and Development in Informaion Retrieval, pp. 267–273 (2003)
86. Zdunek, R., Cichocki, A.: Non-negative matrix factorization with quasi-newton optimization. In: Proceedings of the Eighth International Conference on Artificial Intelligence and, Soft Computing, pp. 870–879 (2006)
87. Zdunek, R., Cichocki, A.: Fast nonnegative matrix factorization algorithms using projected gradient approaches for large-scale problems. Comput. Intell. Neurosci. **2008**, 939567 (2008)
88. Zhong, M., Girolami, M.: Reversible jump MCMC for non-negative matrix factorization. In: Proceedings of the Twelfth International Conference on Artificial Intelligence and Statistics (AISTATS), JMLR: W&CP, vol. 5, pp. 663–670 (2009)